NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

UTILIZATION OF A KALMAN OBSERVER WITH LARGE SPACE STRUCTURES

by

Bruce M. Jackson

December 1988

Thesis Advisor Jeff B. Burl

Approved for public release; distribution is unlimited.

.assified

ty classification of this page

REPORT DOCUMENTATION PAGE

.eport Security Classification Unclassified	1b Restrictive Markings
ecurity Classification Authority	3 Distribution Availability of Report Approved for public release; distribution is unlimited.
Declassification Downgrading Schedule	
rforming Organization Report Number(s)	5 Monitoring Organization Report Number(s)
Name of Performing Organization val Postgraduate School — 6b Office Symbol *(if applicable)* 33	7a Name of Monitoring Organization Naval Postgraduate School
Address *(city, state, and ZIP code)* nterey, CA 93943-5000	7b Address *(city, state, and ZIP code)* Monterey, CA 93943-5000
Name of Funding Sponsoring Organization — 8b Office Symbol *(if applicable)*	9 Procurement Instrument Identification Number
Address *(city, state, and ZIP code)*	10 Source of Funding Numbers Program Element No / Project No / Task No / Work Unit Accession No

Title *(include security classification)* UTILIZATION OF A KALMAN OBSERVER WITH LARGE SPACE STRUCTURES

Personal Author(s) Bruce M. Jackson

Type of Report	13b Time Covered	14 Date of Report *(year, month, day)*	15 Page Count
ster's Thesis	From To	December 1988	85

Supplementary Notation The views expressed in this thesis are those of the author and do not reflect the official policy or po-on of the Department of Defense or the U.S. Government.

Cosati Codes			18 Subject Terms *(continue on reverse if necessary and identify by block number)*
ld	Group	Subgroup	Kalman Observer, large space structures.

Abstract *(continue on reverse if necessary and identify by block number)*

Control of the motions and vibrations of large space structures require the knowledge of state values that may not be ailable due either to inability to measure the states or, the high cost of the sensors to measure the required states. One lution is the use of an observer to estimate the states from limited sensor input.

The physical characteristics of large space structures and the enviroment they operate in will cause large amounts of noise the measurements. The obvious observer for such an enviroment is the Kalman Filter which is specifically designed to oduce optimal estimates in a noisy enviroment.

A straightforward application of the Kalman Filter will be examined utilizing a steady state Kalman gain matrix. The server performance will be examined in both matched filter/plant and reduced order filter configurations.

Distribution Availability of Abstract unclassified unlimited ☐ same as report ☐ DTIC users	21 Abstract Security Classification Unclassified	
a Name of Responsible Individual ff B. Burl	22b Telephone *(include Area code)* (408) 646-2390	22c Office Symbol 62BL

) FORM 1473,84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete — security classification of this page

Unclassified

Approved for public release; distribution is unlimited.

Utilization of a Kalman Observer with Large Space Structures

by

Bruce M. Jackson
Lieutenant Commander, United States Navy
B.S., United States Naval Academy, 1976

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ENGINEERING SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
December 1988

ABSTRACT

Control of the motions and vibrations of large space structures require the knowledge of state values that may not be available due either to inability to measure the states or, the high cost of the sensors to measure the required states. One solution is the use of an observer to estimate the states from limited sensor input.

The physical characteristics of large space structures and the enviroment they operate in will cause large amounts of noise in the measurements. The obvious observer for such an enviroment is the Kalman Filter which is specifically designed to produce optimal estimates in a noisy enviroment.

A straightforward application of the Kalman Filter will be examined utilizing a steady state Kalman gain matrix. The observer performance will be examined in both matched filter/plant and reduced order filter configurations.

THESIS DISCLAIMER

The reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these programs without additional verification is at the risk of the user.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to express my appreciation to the McDonnel Douglas Astronautics Company of Huntington Beach, California, for allowing the use of their dynamic model of a preliminary space station configuration. Also, I would like to thank Major William J. Preston, USMC for allowing the use of the mathematical model for the station dynamics which makes up the majority of Chapter II of this paper as well as portions of his simulation program. Finally, I would like to thank Professor J. B. Burl for his assistance and most of all his patience in the completion of this work.

This paper is dedicated to Christina Marie Jackson (USNA '10).

I. INTRODUCTION

A. BACKGROUND

The advent of large space structures poses a number of problems for the control engineer. Previously, the objects put into space could be treated as rigid bodies so that a single three axis sensor package could be used to tell the motion of all components. The large space structures will not be rigid, instead they will have considerable flexibility and multiple modes of vibration [Ref. 1: p. 51]

Control of the structure's attitude and vibrations requires knowing the motions of the components. One approach would be to heavily instrument the space structure, but weight and cost make this approach impractical. An alternative is to use a limited number of sensors to measure only certain states and to deduce the other required states by use of an observer algorithm.

This thesis will address the production of estimates of the states needed for control of the structure. The model used will be a early design study by McDonnell Douglas Astronautics for a dual keel space station. The techniques and problems of observation for this model are generic to all large space structures.

B. PROBLEM STATEMENT

Design of an observer for estimating the states of a large space structure breaks down into several steps. First, a matematical model is developed for the system behavior over time. Modal analysis is used to form a system composed of decoupled second order differential equations. The use of decoupled equations allows a reduced order model to be generated by truncating the number of modal equations. A reduced order model will have all of the same mathematical qualities (and problems) but reduces the amount of time and computer resources required to do simulation.

Second, the observer is designed. The observer is designed to obtain a minimum variance estimate of the desired state values from the measurements.

Third, the observer is simulated to verify performance. Simulation runs of both a matched observer/plant system and a reduced order observer are employed. That is, the system is run where the observer is used to estimate all of the plant states and run where there are more plant states than the observer estimates.

Fourth, results are analysed and conclusions drawn based on these results. Reccomendations for further areas of research are suggested based on the results and conclusions.

C. ORGANIZATION

The model of the space station is developed in Chapter II. The modal model was developed using modal analysis and discretized to form the discrete-time state equations. The data for this model was from an early design study by McDonnell Douglas Astronautics Company for a dual-keel space station. The observer and its equations are developed in Chapter III. Chapter IV is the simulation runs of the observer versus the plant. Chapter V presents conculsions and recommendations for further research.

II. MATHEMATICAL MODEL

A. INTRODUCTION

Prior to the proposed space station almost all of the objects put into space could be treated as simple rigid bodies for the purpose of mathematical modelling of their motions. The design constraints imposed by the high cost of lifting mass to orbit dictates a light, open structure with considerable flexing. Large space structures such as the space station, therefore, cannot be treated as rigid bodies. The structure is in fact lightly damped with multiple natural frequencies. The result is a structure that will vibrate for considerable periods of time whenever external forces are applied.

The space station structure can be modeled as an n-DOF (degree of freedom) system consisting of n masses, springs, and dashpots [Ref. 2: p. 173-176]. This straight forward modelling of the coupled masses produces a system of unworkable complexity. As a result, the system will be modelled in terms of the structures natural modes of vibration. The resulting system, while still complex, is at least workable.

The model will be developed in two steps. The first will be to generate the continous-time model of the natural modes. The second will yield the discrete-time model, developed from the first model, for use in the simulation.

B. MODAL MODEL

The space station structure can be modeled as a system of discrete masses coupled by springs and dashpots. The major mechanism of damping in the structure is structural damping, the internal dissipation of energy within the members, as the structure vibrates. Structural damping can be shown to be equivalent to viscous damping and this equivalency is used in the model [Ref. 2: p. 72-73].

The energy dissipated by structural damping is:

$$Wd = \alpha X^2 \tag{1}$$

Wd = energy dissipated by structural damping

α = constant (force/displacement)

X = displacement

The energy dissipated by viscous damping is:

$$Wv = \pi c \omega X^2 \tag{2}$$

We can equate the two

$$\pi C_{eq}\omega X^2 = \alpha X^2 \tag{3}$$

yielding an equivalent viscous damping coefficient:

$$C_{eq} = \frac{\alpha}{\pi\omega} \tag{4}$$

The second order differential equation for a single viscously damped mass is:

$$m\ddot{x} + c\dot{x} + kx = F(t) \tag{5}$$

Substituting C_{eq} for c

$$m\ddot{x} + \frac{\alpha}{\pi\omega}\dot{x} + kx = F(t) \tag{6}$$

For multiple mass systems C_{eq} becomes $\frac{d}{\omega_f}\mathbf{K}$ where ω_f is the natural frequency of vibration.

The displacement of masses can be represented by the second order matrix differential equation [Ref. 3: p. 3-9],

$$\mathbf{M}\ddot{q}(t) + \frac{d}{\omega_f}\mathbf{K}\dot{q}(t) + \mathbf{K}q(t) = \mathbf{F}(t) \tag{7}$$

q = coordinate vector

$\mathbf{M}$ = system mass matrix (diagonal)

$\frac{d}{\omega_f}\mathbf{K}$ = equivalent damping

d = damping coefficient

ω_f = frequency of oscilation of the system

$\mathbf{K}$ = symmetric system stiffness matrix

$\mathbf{F}(t)$ = system forcing function

The above equation represents a system of second order differential equations coupled through the stiffness matrix. Decoupling can be done by expressing q in terms of natural modes of vibration. The process is called modal analysis. The independent differential equations can then be treated individually. The modal equations are derived below.

First, the undamped, homogeneous form of Eq. (7)

$$\mathbf{M}\ddot{q}(t) + \mathbf{K}q(t) = 0 \tag{8}$$

is solved. Let

$$q(t) = Ax \sin(\omega t + \Theta) \tag{9}$$

$$\dot{q}(t) = Ax\omega \cos(\omega t + \Theta) \tag{10}$$

$$\ddot{q}(t) = -Ax\omega^2 \sin(\omega t + \Theta) \tag{11}$$

substituting Eq. (9) and Eq. (10) into Eq. (11)

$$[-\omega^2\mathbf{M} + \mathbf{K}]\mathbf{A}x \sin(\omega t + \Theta) = 0 \tag{12}$$

This equation has a non-trivial solution for all time if and only if:

$$[\mathbf{K} - \omega^2\mathbf{M}]x = 0 \tag{13}$$

Equation (12) has n combinations of x (natural mode shapes) and ω (natural frequencies) as solutions. These can be grouped into matrices:

$$\mathbf{X} = [x_1 x_2 \ldots x_n]^T \tag{14}$$

$$\Omega^2 = diag[\omega_{o1}^2 \omega_{o2}^2 \ldots \omega_{on}^2] \tag{15}$$

which satisfy the equation:

$$\mathbf{KX} = \Omega^2\mathbf{MX} \tag{16}$$

Several useful relations can be derived from Eq. (16). Premultiplying Eq. (16) by $\mathbf{X}^T$,

$$\mathbf{X}^T\mathbf{KX} = \Omega^2\mathbf{X}^T\mathbf{MX} \tag{17}$$

The eigenvectors can be normalized

$$\mathbf{X}^T\mathbf{MX} = \mathbf{I} \tag{18}$$

which yields

$$\mathbf{X}^T\mathbf{KX} = \Omega^2 \tag{19}$$

The equations of motion can be uncoupled through the linear transformation of the coordinate system

$$q(t) = \sum_{i=1}^{n} x_i \eta_i(t) = \mathbf{X}\eta(t) \tag{20}$$

X = modal matrix

n = maximum number of degrees of freedom

$\eta(t)$ = transformed coordinate vector

Application of the transformation to the system Eq. (7) yields

$$\mathbf{X}^T\mathbf{M}\mathbf{X}\ddot{\eta}(t) + \frac{d}{\omega_f}\mathbf{X}^T\mathbf{K}\mathbf{X}\dot{\eta}(t) + \mathbf{X}^T\mathbf{K}\mathbf{X}\eta(t) = \mathbf{X}^T\mathbf{F}(t) \tag{21}$$

Using Eq.(18) and Eq. (19)

$$\mathbf{X}^T\mathbf{M}\mathbf{X}\ddot{\eta}(t) = \mathbf{I}\ddot{\eta}(t) = \ddot{\eta} \tag{22}$$

$$\frac{d}{\omega_f}\mathbf{X}^T\mathbf{K}\mathbf{X}\dot{\eta}(t) = \frac{d}{\omega_f}\Omega^2\dot{\eta}(t) = d\Omega\dot{\eta} \tag{23}$$

$$\mathbf{X}^T\mathbf{K}\mathbf{X}\eta(t) = \Omega^2\eta \tag{24}$$

therefore

$$\ddot{\eta} + d\Omega\dot{\eta} + \Omega^2\eta = \mathbf{X}^T\mathbf{F} \tag{25}$$

Equation (25) is the modal model of uncoupled second order differential equations. The motion of the structure can be found from the modal amplitudes, $\eta(t)$, using Eq. (20).

C. DISCRETE-TIME MODEL

The discrete-time state space model is found by solving the continuous-time equations. The *ith* equation of motion is

$$\ddot{\eta}_i(t) + d\omega_{oi}\dot{\eta}_i(t) + \omega_{oi}^2\eta(t) = \mathbf{X}_i^T\mathbf{F}(t) \tag{26}$$

$\mathbf{X}_i^T$ = transpose of the ith mode shape vector

$\mathbf{F}(t)$ = torquing force applied at a point

The homogeneous solution ($\mathbf{F}(t) = 0$) for Eq. (26) is [Ref. 4: p. 475-476]

$$\eta_i(t) = C_1 e^{-\gamma t}\cos(\omega_d t) + C_2 e^{-\gamma t}\sin(\omega_d t) \tag{27}$$

where

$$\gamma = \frac{d\omega_{oi}}{2} \tag{28}$$

$$\omega_d = \sqrt{\omega_{oi}^2 - \gamma^2} \tag{29}$$

The constants in Eq. (27) can be found by taking the derivative

$$\dot{\eta}(t) = (C_2\omega_d - C_1\gamma)e^{-\gamma t}\cos(\omega_d t) - (C_1\omega_d - C_2\gamma)e^{-\gamma t}\sin(\omega_d t) \tag{30}$$

and evaluating at $t = 0$

$$\eta(0) = C_1 \tag{31}$$

$$\dot{\eta}(0) = C_2\omega_d - C_1\gamma \tag{32}$$

Solving for C_1 and C_2

$$C_1 = \eta(0) \tag{33}$$

$$C_2 = \frac{\dot{\eta}(0)}{\omega_d} + \frac{\eta(0)\gamma}{\omega_d} \tag{34}$$

In matrix form

$$\begin{bmatrix} C_1 \\ C_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \dfrac{\gamma}{\omega_d} & \dfrac{1}{\omega_d} \end{bmatrix} \begin{bmatrix} \eta(0) \\ \dot{\eta}(0) \end{bmatrix} \tag{35}$$

Rewriting Eq. (27) and Eq. (30) in matrix form

$$\begin{bmatrix} \eta(t) \\ \dot{\eta}(t) \end{bmatrix} = \begin{bmatrix} e^{-\gamma t}\cos(\omega_d t) & e^{-\gamma t}\sin(\omega_d t) \\ e^{-\gamma t}[\gamma\cos(\omega_d t) + \omega_d\sin(\omega_d t)] & e^{-\gamma t}[\omega_d\cos(\omega_d t) - \gamma\sin(\omega_d t)] \end{bmatrix} \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} \tag{36}$$

Substituting Eq. (35) into Eq. (36), the solution can be written in terms of the initial conditions

$$\begin{bmatrix} \eta(t) \\ \dot{\eta}(t) \end{bmatrix} = \begin{bmatrix} e^{-\gamma t}[\cos(\omega_d t) + \frac{\gamma}{\omega_d}\sin(\omega_d t)] & \frac{1}{\omega_d} e^{-\gamma t}\sin(\omega_d t) \\ -\frac{\omega_o^2}{\omega_d} d^{-\gamma t}\sin(\omega_d t) & e^{-\gamma t}[\cos(\omega_d t) - \frac{\gamma}{\omega_d}\sin(\omega_d t)] \end{bmatrix} \begin{bmatrix} \eta(0) \\ \dot{\eta}(0) \end{bmatrix} \tag{37}$$

Letting

$$\mathbf{X}_i(t) = \begin{bmatrix} \eta_i(t) \\ \dot{\eta}_i(t) \end{bmatrix} \tag{38}$$

and

$$\mathbf{A}_i = \begin{bmatrix} e_{-\gamma t}[\cos(\omega_d t) + \frac{\gamma}{\omega_d}\sin(\omega_d t)] & \frac{1}{\omega_d} e^{-\gamma t}\sin(\omega d t) \\ -\frac{\omega_o^2}{\omega_d} e^{-\gamma t}\sin(\omega_d t) & e^{-\gamma t}[\cos(\omega_d t) - \frac{\gamma}{\omega_d}\sin(\omega_d t)] \end{bmatrix} \tag{39}$$

the solution can be written as

$$\mathbf{X}_i(t) = \mathbf{A}_i(t)\mathbf{X}_i(0) \tag{40}$$

where $\mathbf{A}_i$ is the state transition matrix of the *ith* mode. The non-homogeneous solution is

$$\mathbf{X}_i(t) = \mathbf{A}_i(t)\mathbf{X}_i(0) + \mathbf{B}_i x_i^T \mathbf{F}(0) \tag{41}$$

where the discrete-time input matrix, for constant **F**, is given by

$$\mathbf{B}_i = \int_0^t \mathbf{B}_i(\tau)\Gamma\partial\tau \tag{42}$$

and $\Gamma = [0\ 1]^T$ is the input matrix for the continuous-time system, and T is the sampling time. Solving Eq. (42) yields

$$\mathbf{B}_i = \begin{bmatrix} \frac{1}{\omega_o^2}[1 - e^{-\gamma t}\cos(\omega_d t) - \frac{\gamma}{\omega_d} e^{-\gamma t}\sin(\omega_d t)] \\ \frac{1}{\omega_d} e^{-\gamma t}\sin(\omega_d t) \end{bmatrix} \tag{43}$$

The discrete-time state equation for the ith equation of motion can be written as

$$\mathbf{X}_i(kT+1) = \mathbf{A}_i(T)\mathbf{X}_i(kT) + \mathbf{B}_i(T)x_i^T\mathbf{F}(kT) \tag{44}$$

where $\mathbf{A}_i$ and $\mathbf{B}_i$ are evaluated at $t = T$. Here,

$\mathbf{X}_i$ = vector of the ith modal amplitude and the ith modal velocity

$\mathbf{A}_i$ = ith state transition matrix

$\mathbf{B}_i$ = ith input vector

x_i^T = transpose of the ith mode shape vector

$\mathbf{F}$ = distrubuted force on the plant

T = sampling time

k = time index

Equation (44) can be expanded to include the disturbance input, $w(kT)$:

$$\mathbf{X}_i(kT+1) = \mathbf{A}_i(T)\mathbf{X}_i(kT) + \mathbf{B}_i(T)x_i^T[\mathbf{F}(kT) + w(kT)] \tag{45}$$

Equation (45) is the discrete-time mathematical model describing the motion of the structure in terms of its natural modes of vibration.

III. THE OBSERVER

A. INTRODUCTION

The observer design will be required to estimate the modal states in a noisy enviroment. Kalman filtering is the most widely used technique for accomplishing the production of state estimates in a noisy enviroment [Ref. 5: p.159]. The steady state Kalman filter was selected to minimize the computations during the actual plant observer operation. Use of a steady-state gain matrix for the observer allows the matrix to be computed separately from the operational observer, reducing the computer power required for the observer and allowing the algorithm to operate more rapidly.

B. KALMAN FILTER EQUATIONS

The discrete Kalman filter provides state estimates for the following dynamic sytem [Ref. 5: p. 159-162],

$$\mathrm{X}(k+1) = \mathbf{A}\mathrm{X}(k) + \mathbf{B}\mathrm{U}(k) + \mathbf{Bn}\mathrm{W}(k) \tag{46}$$

$$\mathrm{Y}(k+1) = \mathbf{C}\mathrm{X}(k+1) + \mathrm{V}(k+1) \tag{47}$$

X = n x 1 state vector

U = P x 1 control vector

W = r x 1 plant noise vector

Y = m x 1 measurement vector

V = m x 1 measurement noise vector

A = n x n state transition matrix

B = n x p control input matrix

Bn = n x r plant noise input matrix

C = m x n measurement matrix

The plant noise vector $\mathrm{W}(k)$ is gaussian white noise with

$$\mathrm{E}\{\mathrm{W}(k)\} = \mathbf{0} \tag{48}$$

$$\mathrm{E}\{\mathrm{W}(k)\mathrm{W}^T(k)\} = \mathbf{Q} \tag{49}$$

for all $k = 0,1,2,\ldots$, and **Q** is a positive semi-definite r x r matrix. $V(k)$ is gaussian white noise with

$$E\{V(k)\} = 0 \tag{50}$$

$$E\{V(k)V^T(k)\} = \mathbf{R} \tag{51}$$

for all $k = 0,1,2,\ldots$, and **R** is a positive definite m x m matrix. The two random processes $W(k)$ and $V(k)$ are assumed to be independent, so that

$$E\{V(j)W(k)\} = 0 \tag{52}$$

for all $j = 1,2,\ldots$, and $k = 0,1,2,\ldots$ The intial state $X(0)$ is assumed to be a gaussian random vector with

$$E\{X(0)\} = 0 \tag{53}$$

It is assumed that $X(0)$ is independent of $W(k)$ and $V(k)$.

The optimal estimate of $X(k+1)$ is denoted $\hat{X}(k+1 \mid k+1)$. The Kalman filter is designed to minimize

$$J = E\{[X(k+1) - \hat{X}(k+1 \mid k+1)]^T[X(k+1) - \hat{X}(k+1 \mid k+1)]\} \tag{54}$$

The recursive realtions for generating $\hat{X}(k+1 \mid k+1)$ are

$$\hat{X}(k+1 \mid k) = \mathbf{A}\hat{X}(k \mid k) + \mathbf{B}U(k) \tag{55}$$

$$\hat{X}(k+1 \mid k+1) = \hat{X}(k+1 \mid k) + G(k+1)[Y(k+1) - \mathbf{C}\hat{X}(k+1 \mid k)] \tag{56}$$

for $k = 0,1,2,\ldots$, where $\hat{X}(0 \mid 0) = \mathbf{0}$. $\hat{X}(0 \mid 0)$ is set equal to zero since the expectation of $X(0)$ is zero.

$G(k+1)$ is an n x m matrix , called the Kalman Gain Matrix which is specified by the realtions:

$$\mathbf{P}(k+1 \mid k) = \mathbf{AP}(k \mid k)\mathbf{A}^T + \mathbf{BnQ}(k)\mathbf{Bn}^T \tag{57}$$

$$G(k+1) = \mathbf{P}(k+1 \mid k)\mathbf{C}^T[\mathbf{CP}(k+1 \mid k)\mathbf{C}^T + \mathbf{R}(k+1)]^{-1} \tag{58}$$

$$\mathbf{P}(k+1 \mid k+1) = [\mathbf{I} - G(k+1)\mathbf{C}]\mathbf{P}(k+1 \mid k) \tag{59}$$

$P(k \mid k)$ is the covariance matrix of the error between the states and their estimates

$$P(k \mid k) = E\{[X(k) - \hat{X}(k \mid k)][X(k) - \hat{X}(k \mid k)]^T\} \tag{60}$$

Since we are using the steady state gains the choice of $P(0 \mid 0)$ is irrelevant. $P(0 \mid 0)$ is initialized to zero in the gain derivation program for simplicity [Ref. 6: p. 139-140].

C. STEADY-STATE SOLUTION

If Equations (57), (58), and (59) are repeatedly iterated, $G(k+1)$ will converge to a steady state value [Ref. 7: p. 263].

$$G_{ss} = \lim_{k \to \infty} G(k+1) \tag{61}$$

The values of G_{ss} (or G) can be substituted into Eq. (56) making the steady state Kalman filter

$$\hat{X}(k+1 \mid k) = A\hat{X}(k \mid k) + BU(k) \tag{62}$$

$$\hat{X}(k+1 \mid k+1) = \hat{X}(k+1 \mid k) + G[Y(k+1) - C\hat{X}(k+1 \mid k)] \tag{63}$$

D. OBSERVER PERFORMANCE

The performance of an observer is judged by how accurately and rapidly it estimates the desired states. The performance measure of the observer as a whole is shown in equation (54). The normalized performance of the observer for individual states is

$$J_i = E[(x_i - \hat{x}_i)^2]/E[x_i^2] \tag{64}$$

which can be found using Eq. (65)

$$J_i = \sum_{k=0}^{\infty} (x_i(k) - \hat{x}_i(k))^2 T_s \div \sum_{k=0}^{\infty} x_i^2(k) T_s \tag{65}$$

J_i = performance measure for the *ith* state

$x_i(k)$ = value of the *ith* state at k

$\hat{x}_i(k)$ = observer estimateof *ith* state at k

T_s = sample interval

A normalized performance measure is used to aid comparison of the performance of the observer in estimating various states. From Eq. (65) it can be shown that if $\hat{x}_i(k) = 0$ for all $k = 0,1,2,...$ that $\mathbf{J}_i$ would be unity. Therefore, the better the performance of the observer, the smaller the fraction of one $\mathbf{J}_i$ will be.

IV. SIMULATION

A. INTRODUCTION

The objectives of the simulation were to

- determine the sensitivity of the observer performance and settling time to changes in the ratio of plant noise to measurement noise,
- determine the effect on observer performance and settling time of increasing the number of modes observed in the matched plant/observer, and
- determine the performance for the reduced order observer.

B. PLANT AND OBSERVER DATA

The dynamic model is a truncated form of a preliminary space station configuration; the phase II dual keel structure.[1] The full model consists of an infinite number of natural modes but this was restricted to the first ten active modes for this study due to limitations on computer resources. As will be shown reasonable data can be obtained with this simplification in examining the observer performance.

C. SIMULATION PROGRAMS

The simulation was broken down into two segments due to the large memory and computational time requirements. The first program computed the steady state observer gain matrix (G). The second program ran the observer and the plant when the plant was subjected to an impulse excitation.

The steady state observer gain matrix (G) was obtained by repeated iteration of equations (57), (58), and (59). The equations were were run until the values of the matrix changed by less than a set fraction. The following formula was used to check the changes in the gain matrix elements

$$\Delta g_{i,j} = [g_{i,j}(k+1) - g_{i,j}(k)]/g_{i,j}(k+1) \tag{66}$$

The program was terminated when $\delta g_{i,j}$ was less than 10^{-10} .

The settling time for the estimates of the states to be within 2% of the actual states was determined by finding the eigenvalues of A - G*C then computing as follows [Ref. 6: p. 139-143]

---

1 The model for preliminary station configuration was provided courtesey of McDonnel Douglas Astronautics Company, 5301 Bolsa Avenue, Huntington Beach, CA 92647.

$$T_s = \log(.02)/\log(\lambda_{AGC_{min}}) \qquad (67)$$

The expected error in the sensor, i.e., the standard deviation of the noise in the measurement, was choosen as 10^{-3} feet per sec. per sec. based on the natural frequencies in the structure and reasonable sensor sensitivity [Ref. 2: p. 79-80]. The expected plant noise was varied to find the range of ratios between plant and measurement noise that the filter would be effective. This approach was taken since the plant noise contributors are not currently well defined.

The second program subjected the plant as modeled in Eq. (45) to an impulse excitation and then had the observer estimate the selected states using observer equations (62) and (63). Observer performance was computed using Eq. (65).

A third program was used to find the contribution of unobserved modes to the noise in the Kalman observer. The program ran the plant subject to an impulse excitation and computed the product of the measurement matrix **C** times the unobserved modes of the state vector $X(k)$ for a measure of the noise contributed by the unobserved modes.

The three programs are listed in the appendices.

D. EFFECT OF PLANT TO MEASUREMENT NOISE RATIO ON OBSERVER PERFORMANCE

The ratio of the variance of the plant noise (PN) to the variance of the measurement noise (MN) was found to have a strong effect on the Kalman Observer performance (J) and settling time (T_s) . Figures 1 through 6 show the observer performance for a 3 mode matched plant and observer system for progressively smaller PN/MN ratios.[2] Figure 7 shows the performance for the seventh mode (position) versus several values of PN/MN. Figure 8 is the settling time versus the same PN/MN ratios.

The figures show that, for all of the plotted performance values, the observer performance is at least marginally acceptable regardless of the PN/MN ratio. Decreasing the PN/MN ratio leads to an even more rapid degradation in observer performance. The settling times also rapidly increase as the PN/MN ratio decreases.

---

2 Figures 1-6 and 9-15 show the performance measure for each mode. The bar for the mode position is immeadiately to the right of the numbered tick mark on the x-axis scale, the mode velocity is next to it immeadiately adjacent to the tick mark without a number.

E. EFFECTS OF INCREASED MODES ON OBSERVER PERFORMANCE

The matched plant/observer was run with increasing numbers of modes to see if there was an effect on observer performance (J) or settling time T_s. Figures 7 through 17 are of observer performance for systems with increasing numbers of modes in the system being observed. Figure 18 is of settling time versus the number of modes in the system. The ratio of PN/MN was kept constant at PN/MN = 2.5×10^9.

The increasing of the number of modes for the matched plant/observer had neglible effect on the performance for the individual modes. The performance value for the modes was effectively constant. Settling times for the observers increased as the number of modes was increased.

F. REDUCED ORDER KALMAN OBSERVER

The Kalman Observer has been shown to be effective where the number of modes observed matches the number of modes in the plant. The Kalman Observer was then run with the one less mode observed than the number of modes in the plant. The gain matrix (G) from the matched system was used. The observer failed with the state estimates produced by the observer becoming excessively large and having settling times of hours vice minutes. Since the purpose of the observer was to provide estimates for use in controlling the plant the time delay makes the estimates unusable.

The cause of the observer failure is apparent when you look at the last portion of Eq. (56) of the Kalman Observer

$$\mathbf{G}[\mathbf{Y}(k+1) - \mathbf{C}\hat{\mathbf{X}}(k+1 \mid k)] \tag{68}$$

This portion of the observer equation is the correction of $\hat{X}(k+1 \mid k)$ to produce $\hat{X}(k+1 \mid k+1)$. The design of the Kalman observer is to produce an estimate despite the measurement noise but, with the reduced order filter there is additional unanticipated noise which causes over correction of the values of $\hat{X}$ leading to the state estimates being excessively large and settling times being too long. This can be shown by examining what composes $Y(k+1) - CX(k+1 \mid k)$

$$
\mathbf{C}\begin{bmatrix} x_1(k) \\ x_2(k) \\ \uparrow \\ \downarrow \\ x_{m-1}(k) \\ x_m(k) \\ \text{---} \\ x_{m+1}(k) \\ x_{m+2}(k) \\ \uparrow \\ \downarrow \\ x_{n-1}(k) \\ x_n(k) \end{bmatrix} - \mathbf{C}\begin{bmatrix} \hat{x}_1(k) \\ \hat{x}_2(k) \\ \uparrow \\ \downarrow \\ \hat{x}_{m-1}(k) \\ x_m(k) \\ \text{---} \\ 0 \\ \uparrow \\ \downarrow \\ 0 \end{bmatrix} \tag{69}
$$

C times the state $x_{m+1}(k)$ through $x_n(k)$ is unanticipated noise so if

$$
\mathbf{C}\begin{bmatrix} x_1(k) \\ x_2(k) \\ \uparrow \\ \downarrow \\ x_{m-1}(k) \\ x_m(k) \end{bmatrix} - \mathbf{C}\begin{bmatrix} \hat{x}_1(k) \\ \hat{x}_2(k) \\ \uparrow \\ \downarrow \\ \hat{x}_{m-1}(k) \\ \hat{x}_m(k) \end{bmatrix} = 0 \tag{70}
$$

the remaining portion of the C matrix times the modal states is an equivalent noise.

Table (1) shows the growth of the unanticipated noise in the filter as the number of unobserved modes in the plant grows. Table (2) shows the individual contributions of the individual modes when left unobserved. Table (1) shows that the unanticipated noise is much larger than that expected by the filter (10^{-3}). Table (2) shows that there are modes that do not markedly contribute to the noise and that they might successfully be left unobserved if the measurement noise estimate was already much larger than these noise sources.

Table 1. CUMULATIVE UNANTICIPATED NOISE FROM UNOBSERVED MODES

Number of Unobserved Modes	Unobserved Modes	E1	E2	E2
1	10	0.647	97.440	3.277
2	10-11	366.354	95.764	3.142
3	10-12	366.355	95.764	3.142
4	10-13	365.565	95.855	3.143
5	10-14	365.426	96.032	5.704
6	10-15	144195.7	116.205	2201.94
7	10-16	148006.8	170.142	5475.21
8	10-17	473974.3	39424.1	5692.50
9	10-18	474344.2	60078.8	9419.77
10	10-19	474358.5	68987.7	9865.27

Table 2. UNANTICIPATED NOISE FROM UNOBSERVED MODES BY MODE

Unobserved Mode	E1	E2	E3
10	0.64716	97.4403	3.27761
11	359.342	0.20929	0.21429
12	0.9353E-08	0.1364E-07	0.2196E-07
13	0.3852E-02	0.4409E-02	0.9017E-03
14	0.5615E-03	0.2592E-01	2.57554
15	143090.9	17.9682	2167.02
16	195.736	83.3458	5675.91
17	324471.2	39252.26	221.170
18	216.240	21133.6	3736.93
19	7.69298	8829.95	458.504
20	2.3194	3949.16	108.981

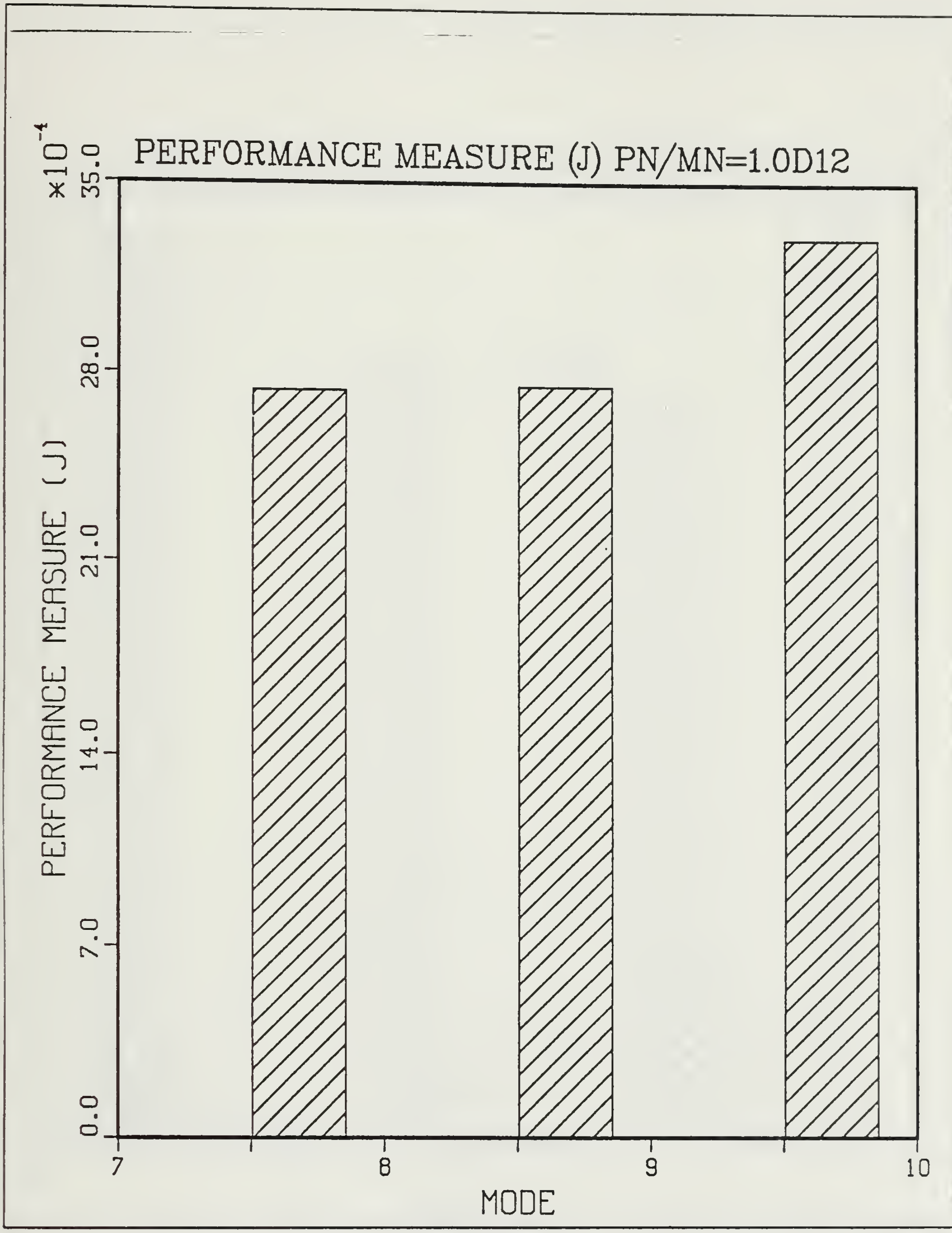

Figure 1. Observer Performance (J) PN/MN = 1.0d12

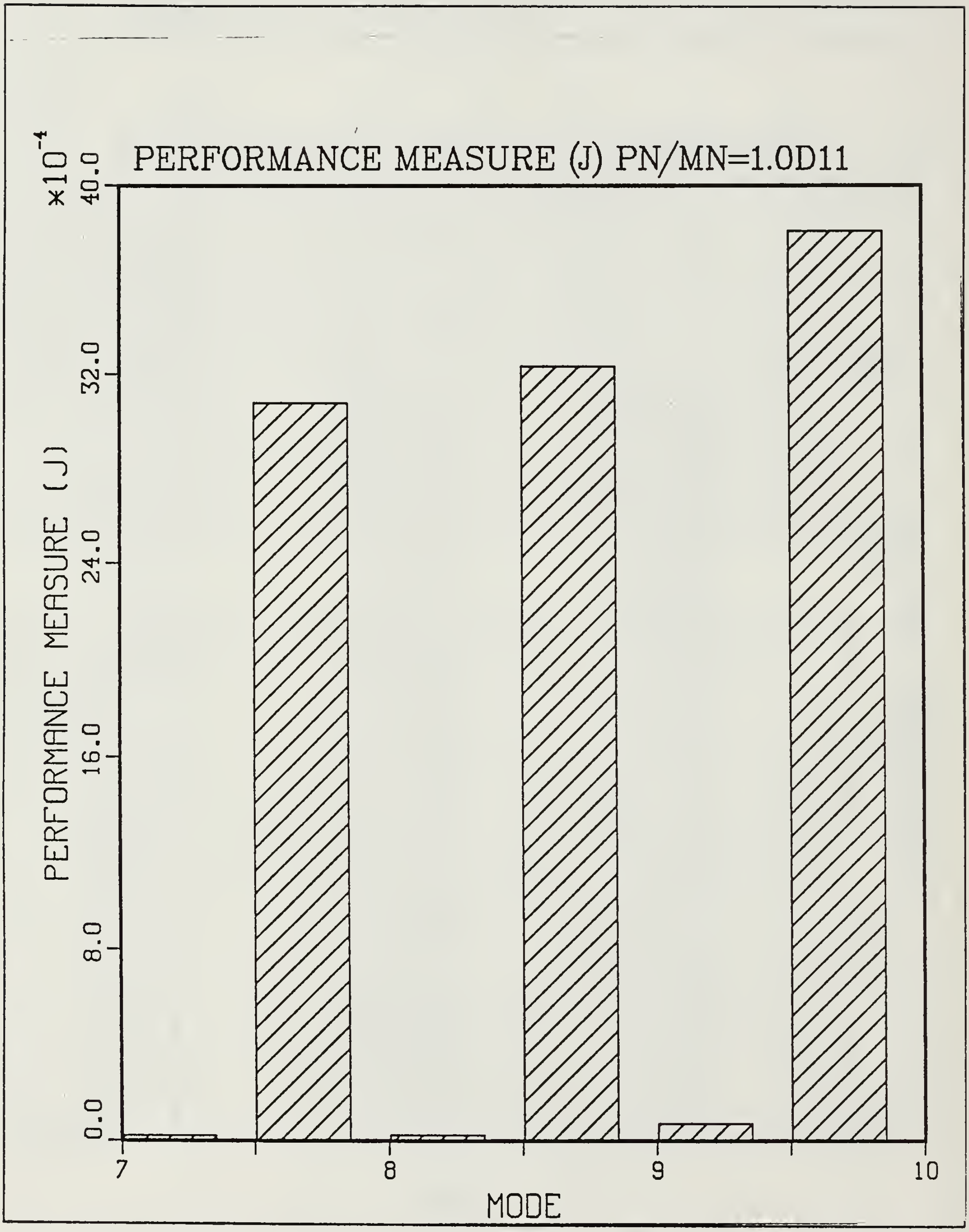

Figure 2. Observer Performance (J) PN/MN = 1.0d11

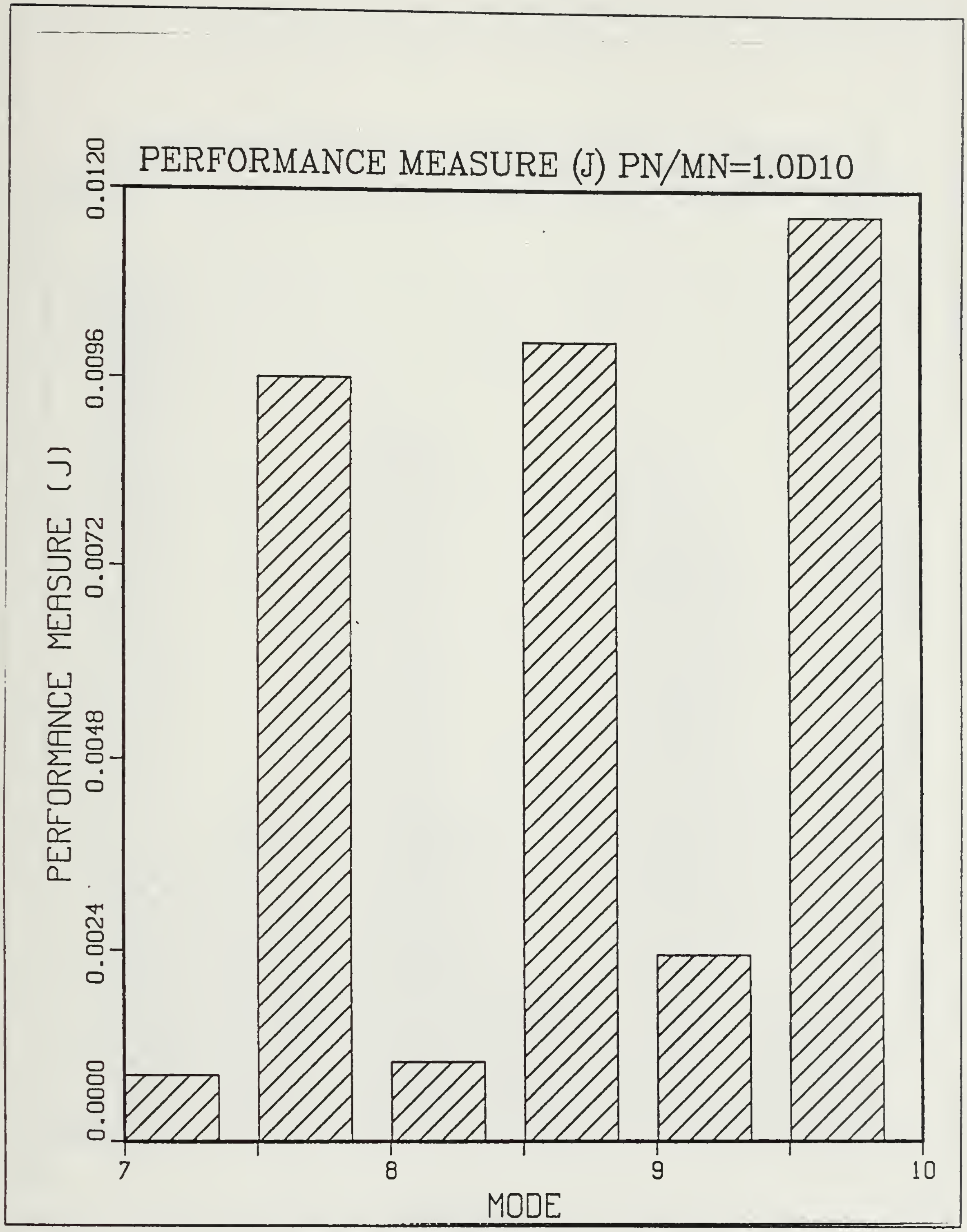

Figure 3. Observer Performance (J) PN/MN = 1.0d10

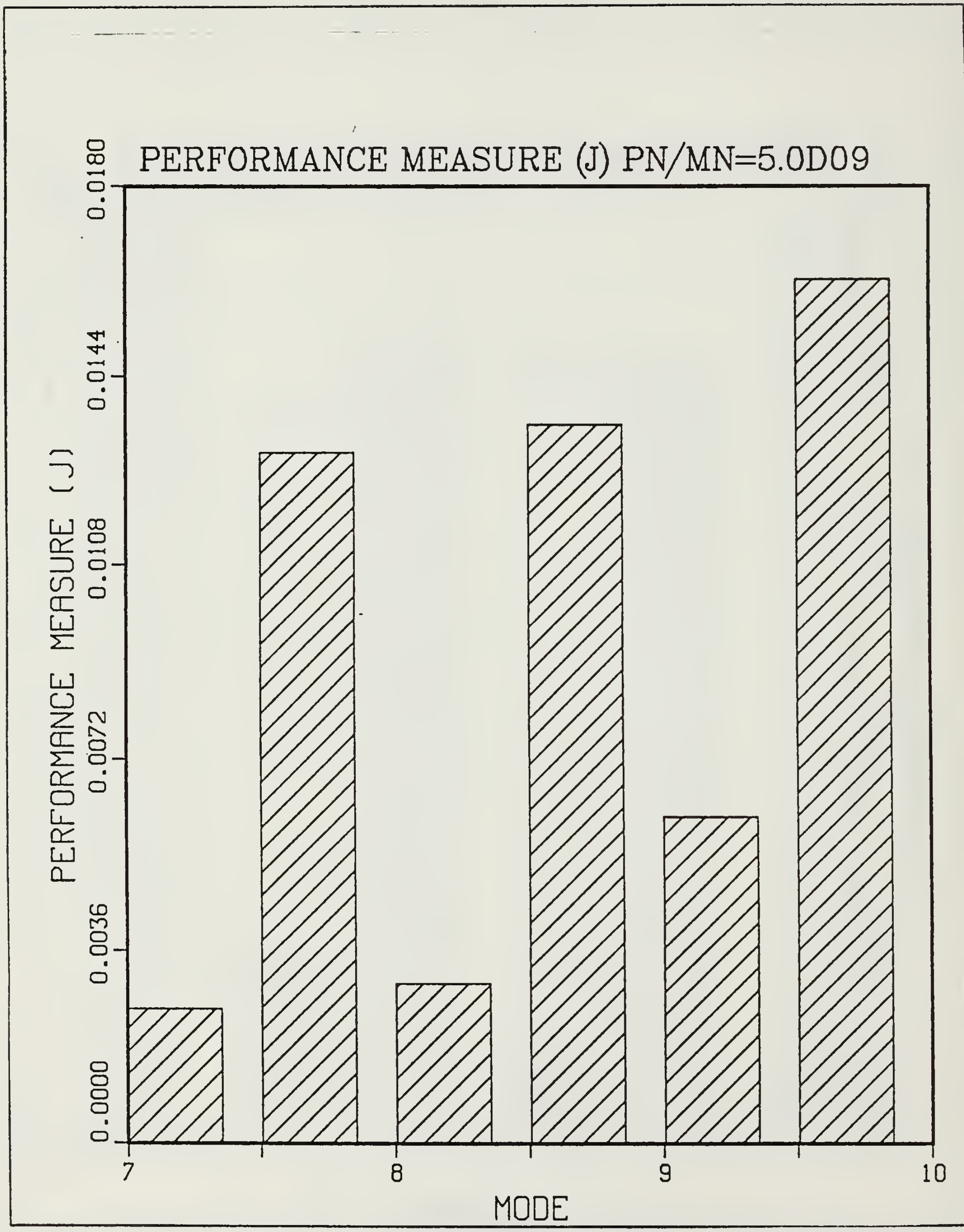

Figure 4. Observer Performance (J) PN/MN = 5.0d09

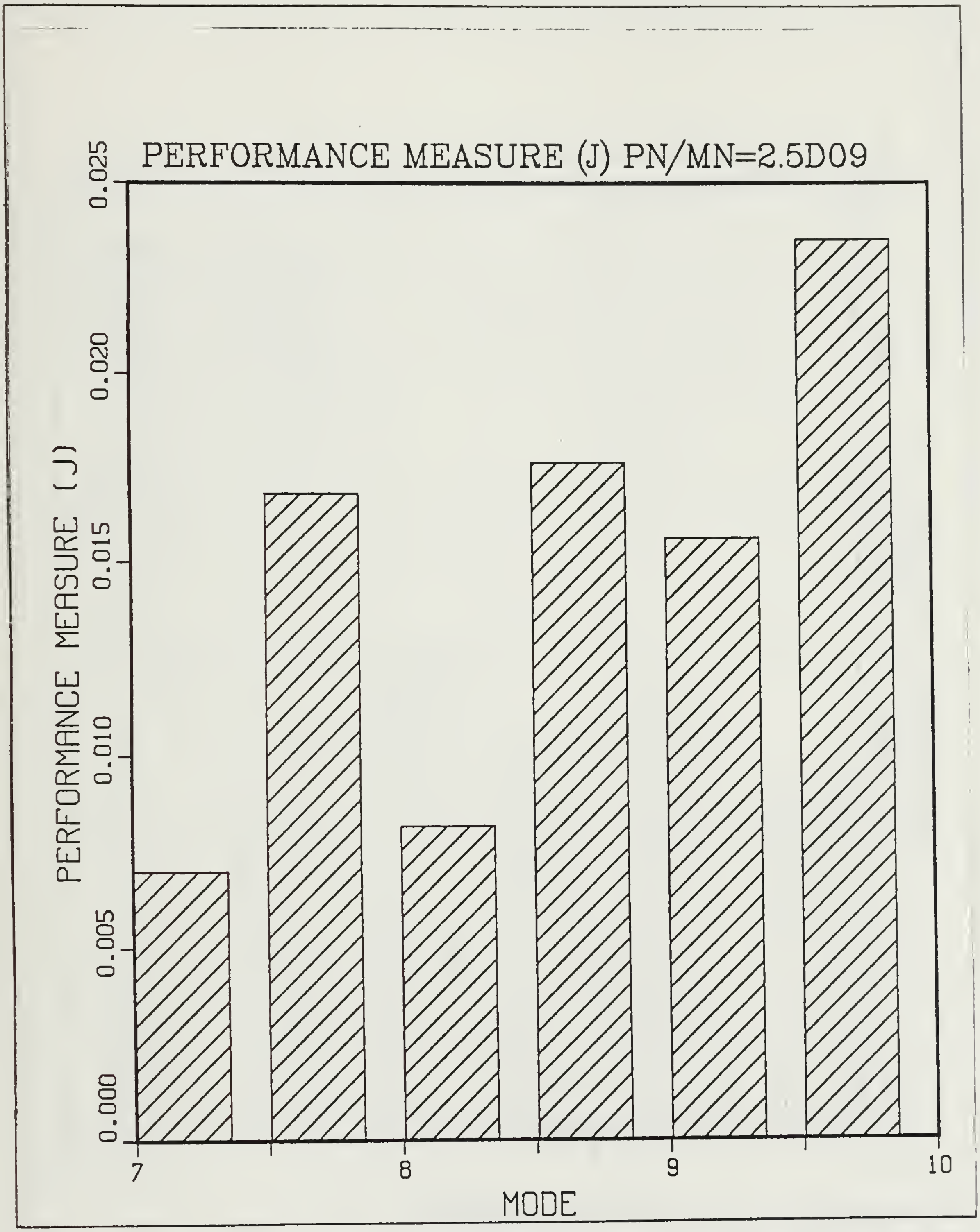

Figure 5. Observer Performance (J) PN/MN = 2.5d09

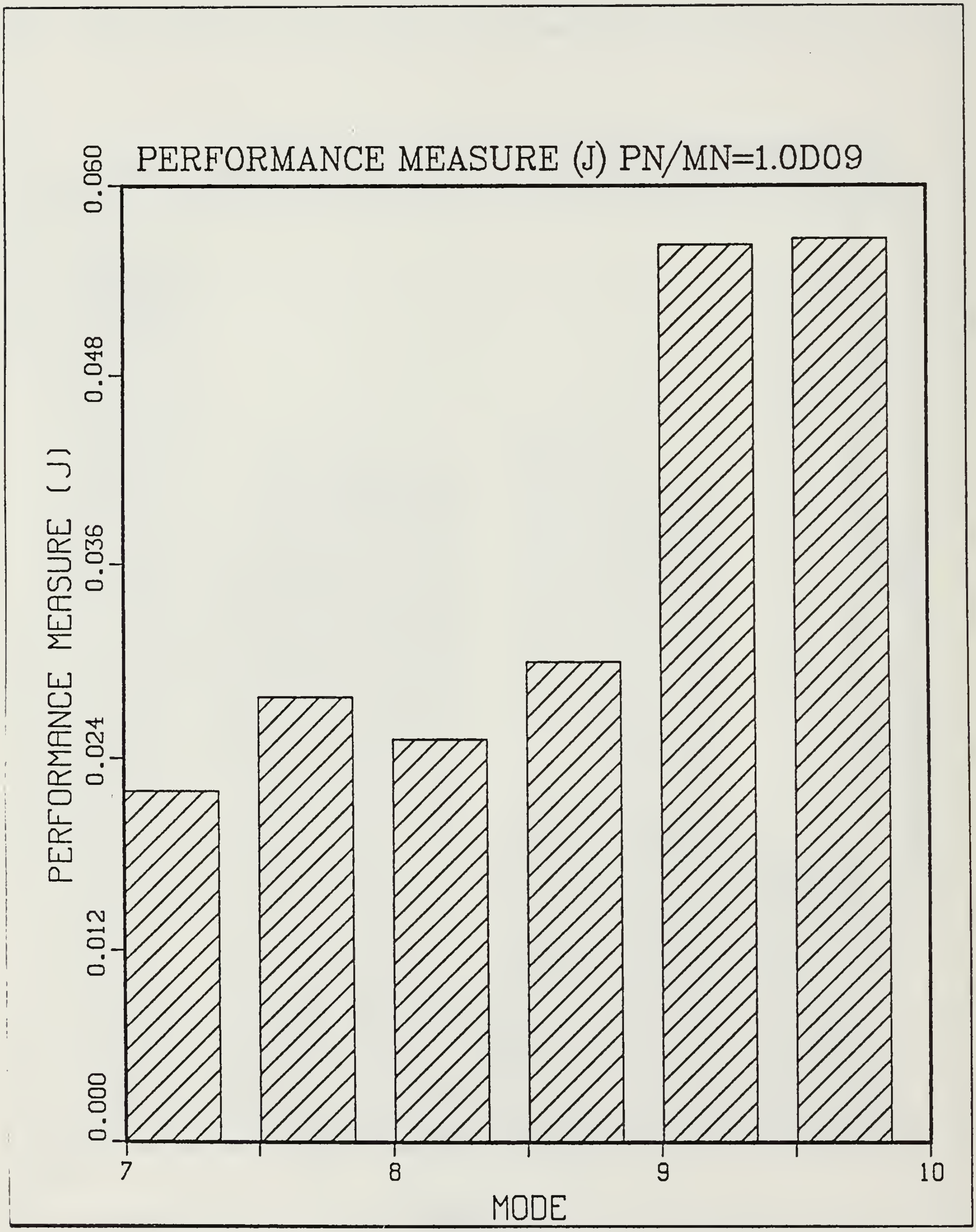

Figure 6. Observer Performance (J) PN/MN = 1.0d09

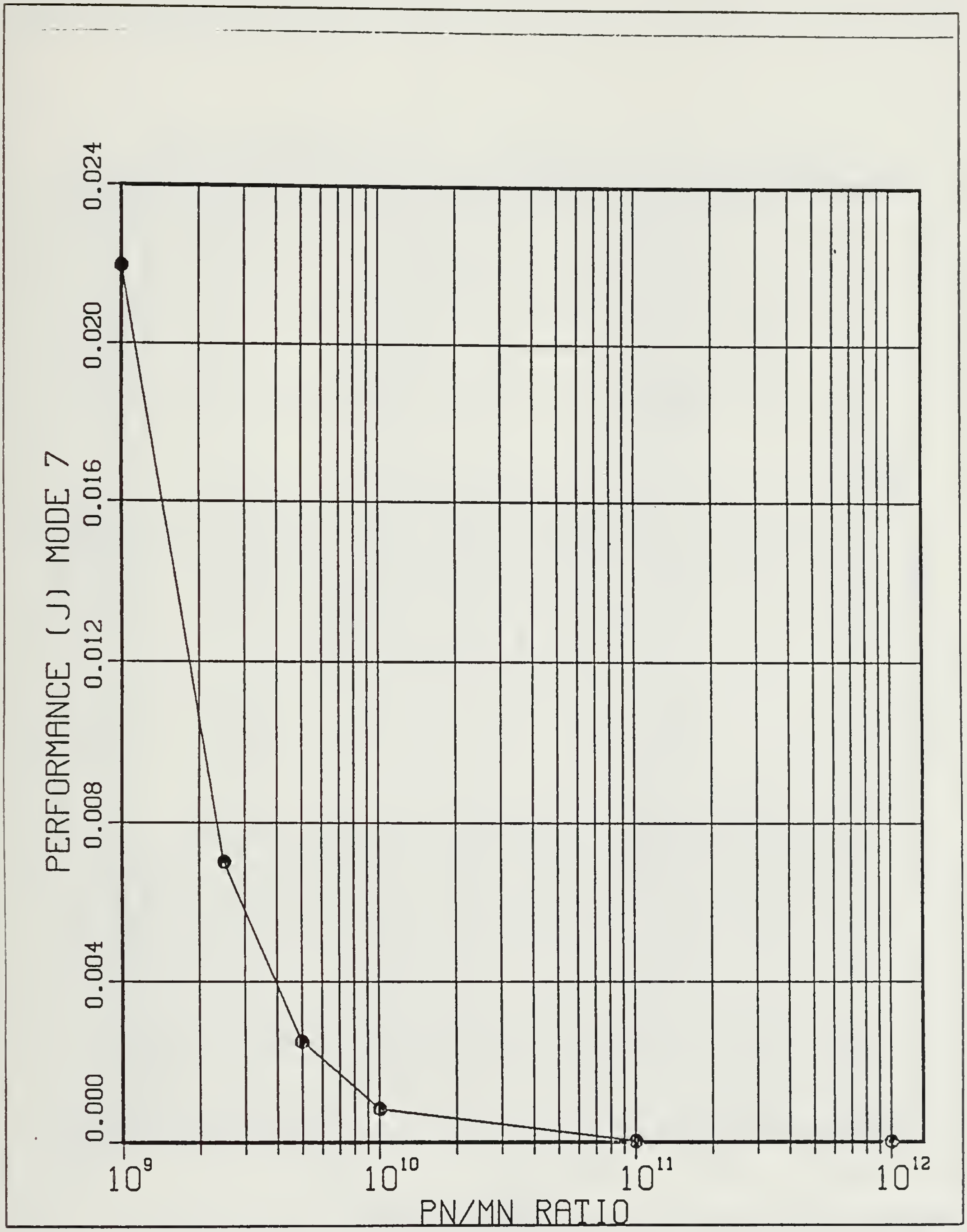

Figure 7. Mode 7 (Postion) Observer Performance versus PN/MN

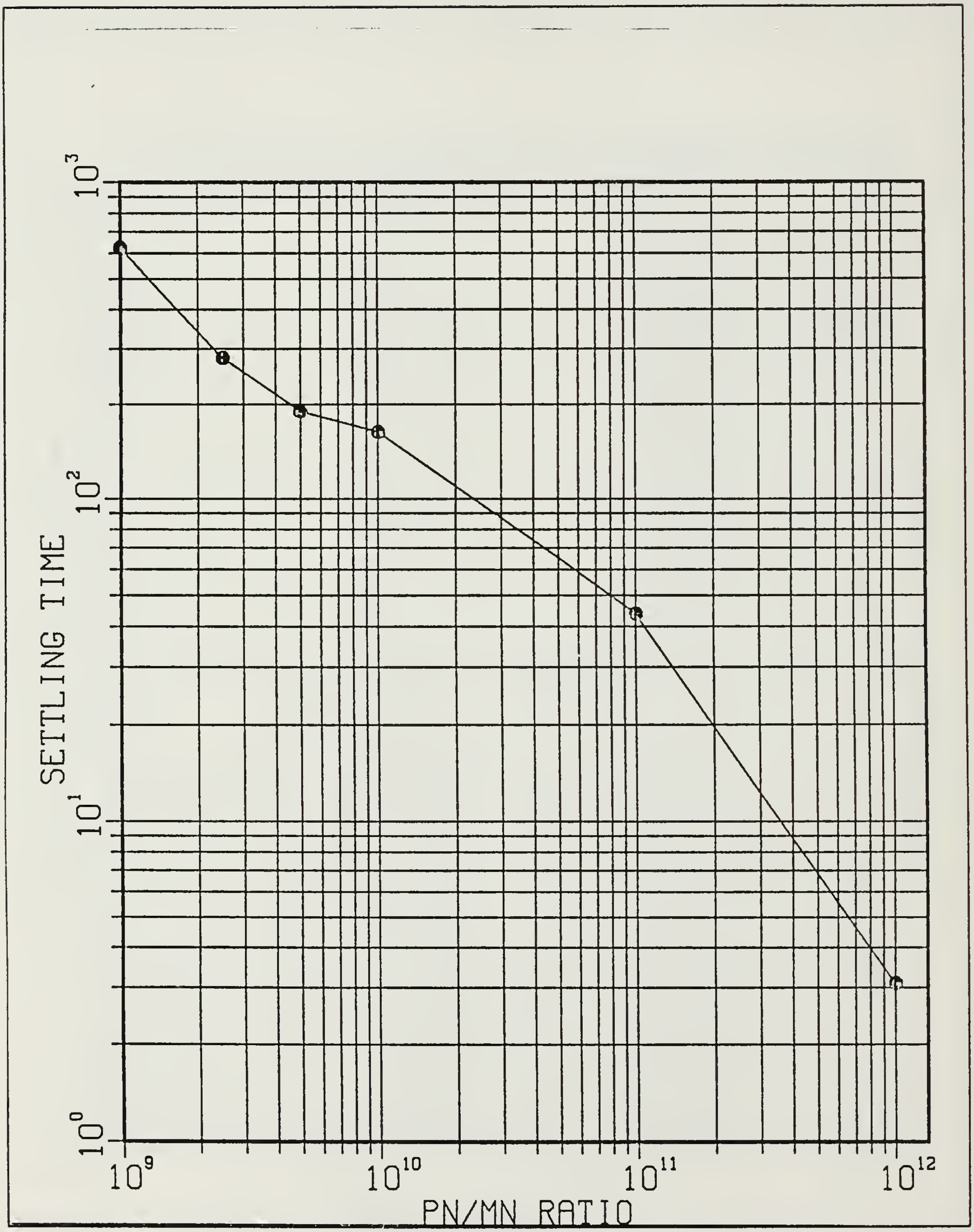

Figure 8. Settling Time versus PN/MN

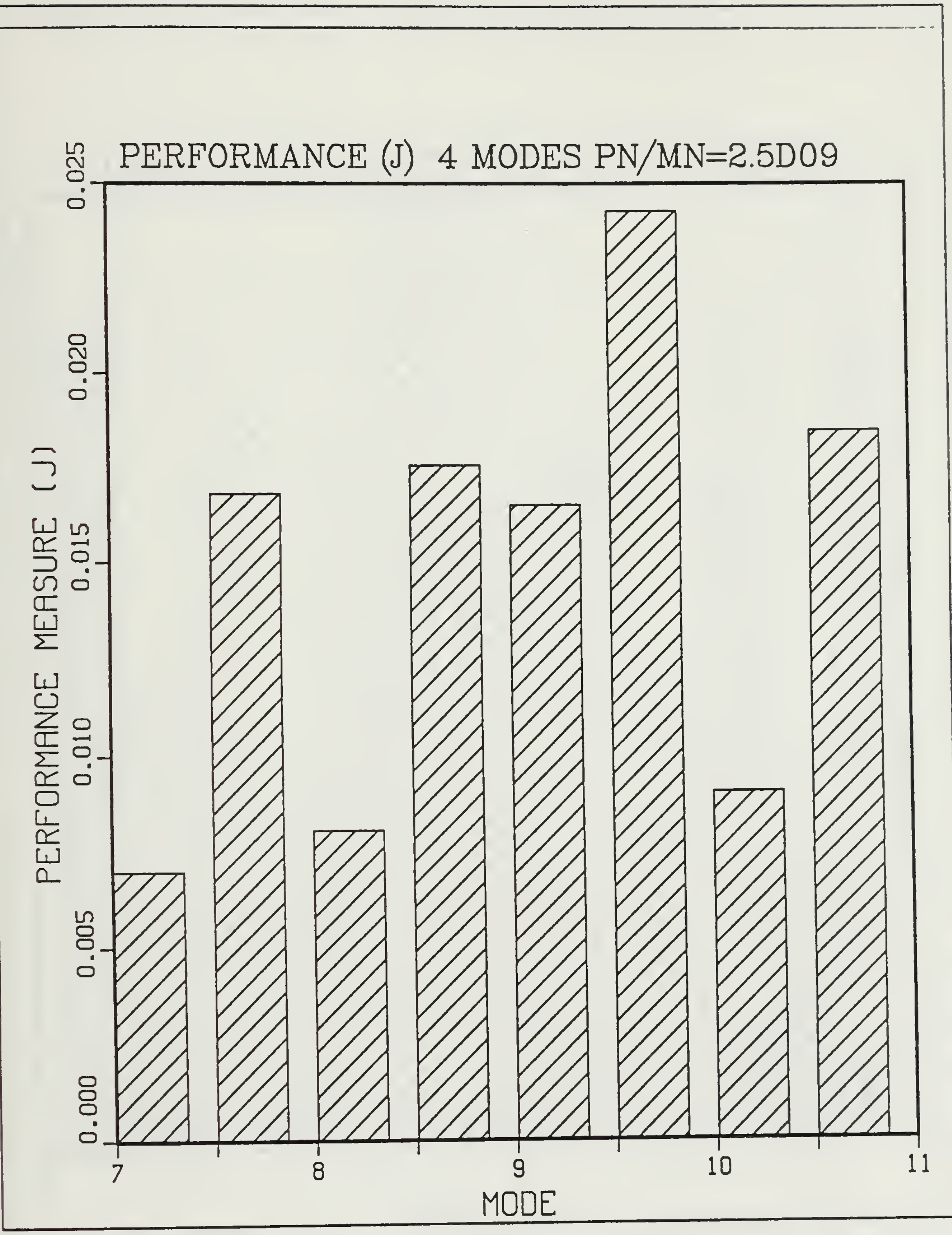

Figure 9. Observer Performance (J) 4 Modes (7 - 10)

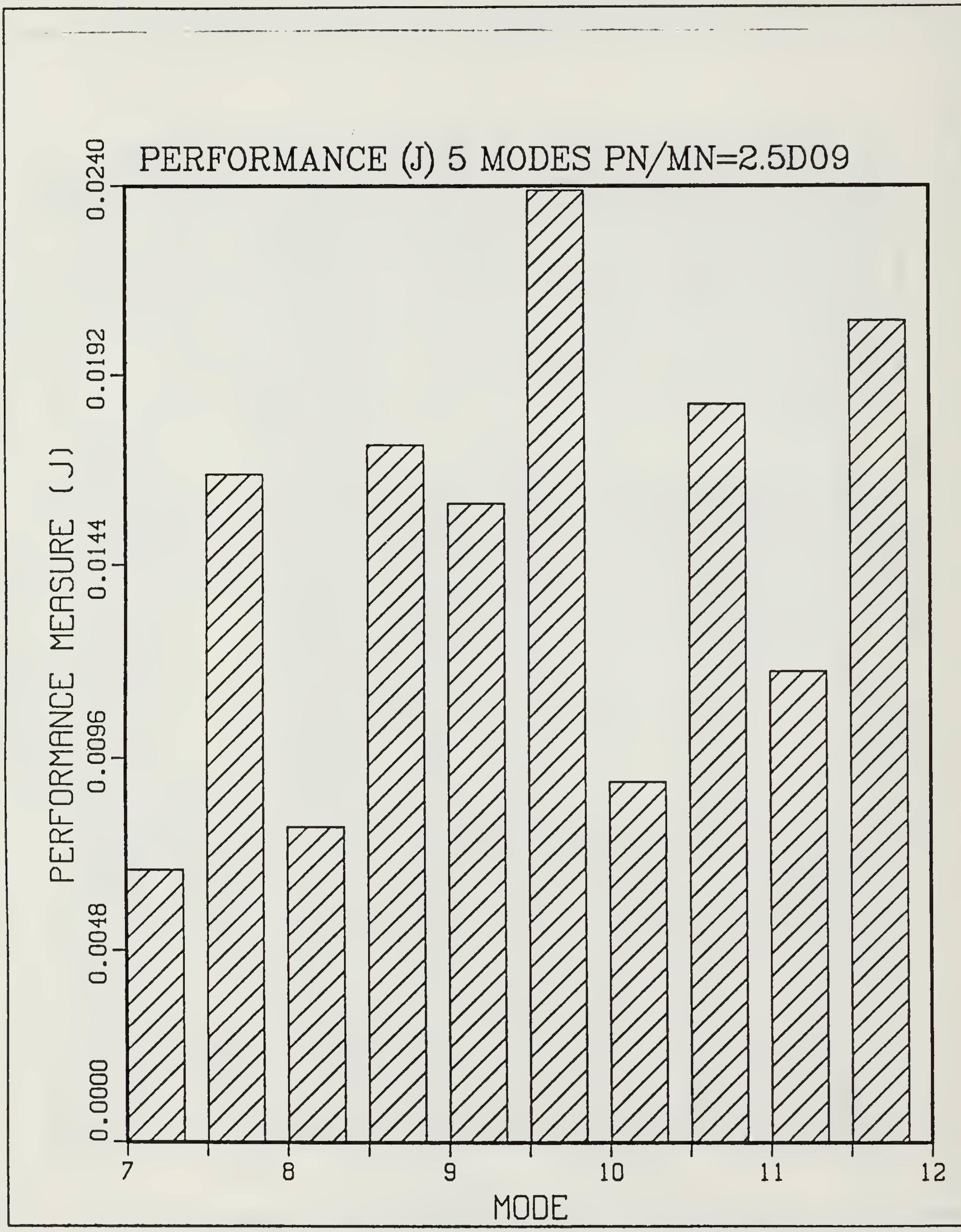

Figure 10. Observer Performance (J) 5 Modes (7 - 11)

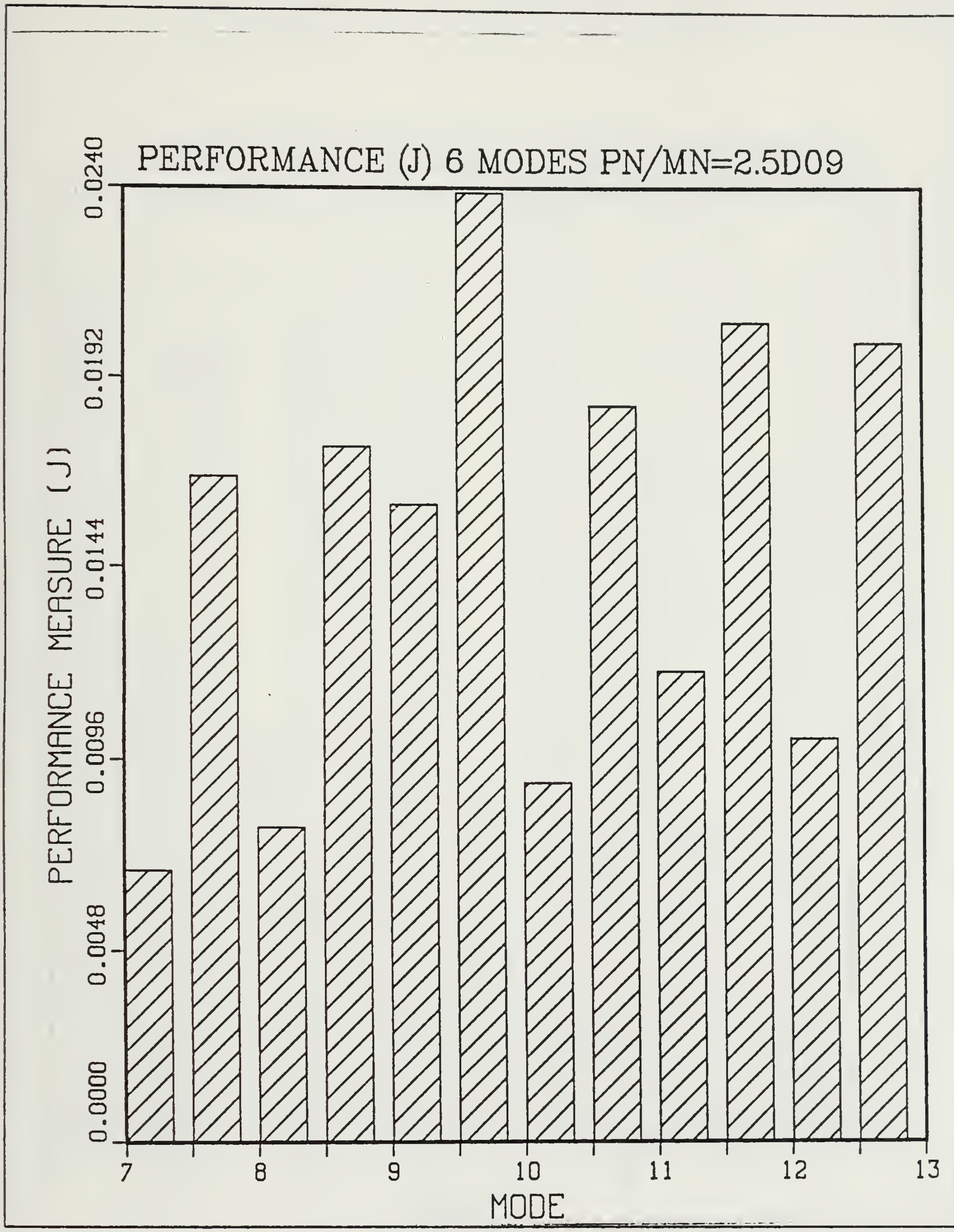

Figure 11. Observer Performance (J) 6 Modes (7 - 12)

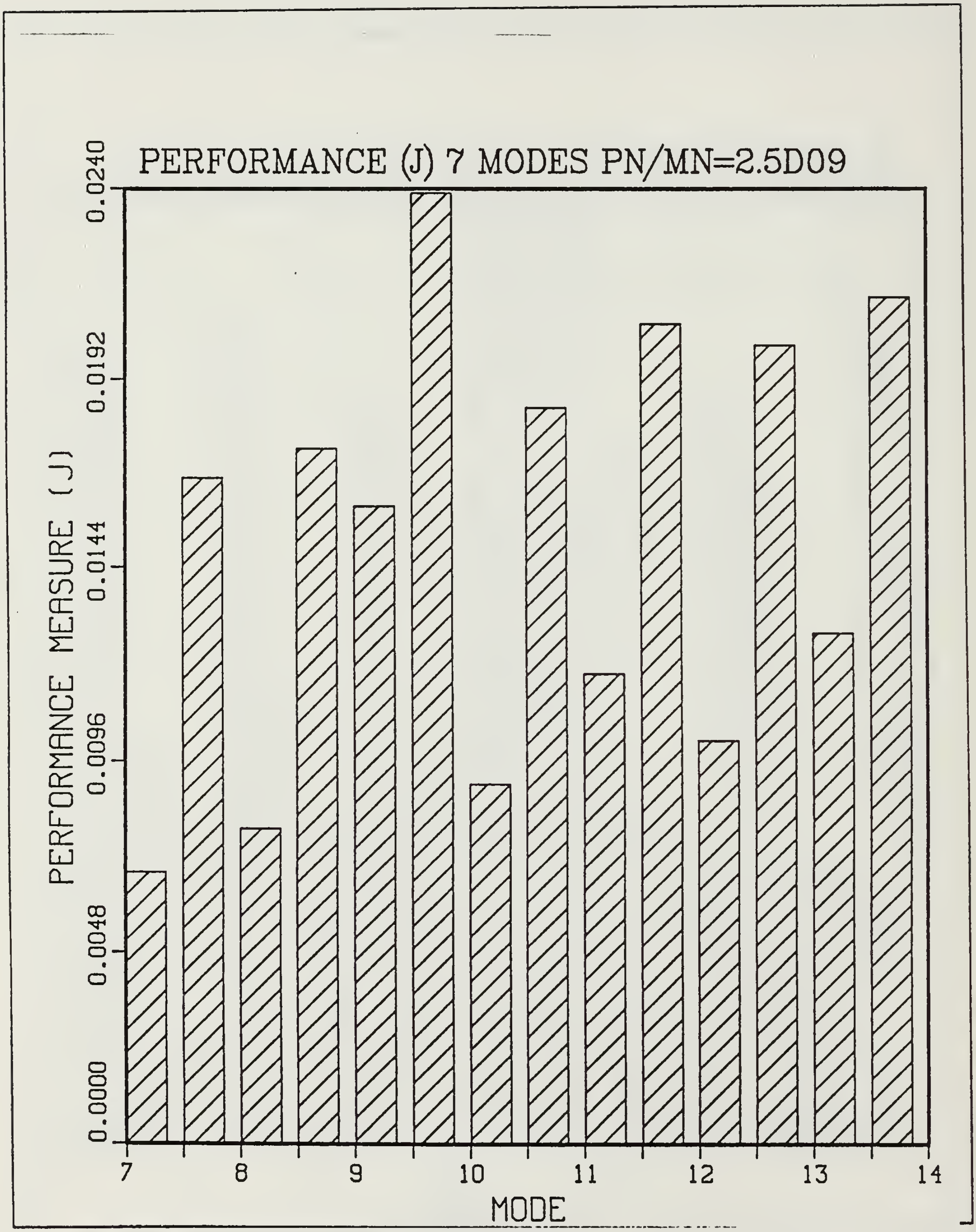

Figure 12. Observer Performance (J) 7 Modes (7 - 13)

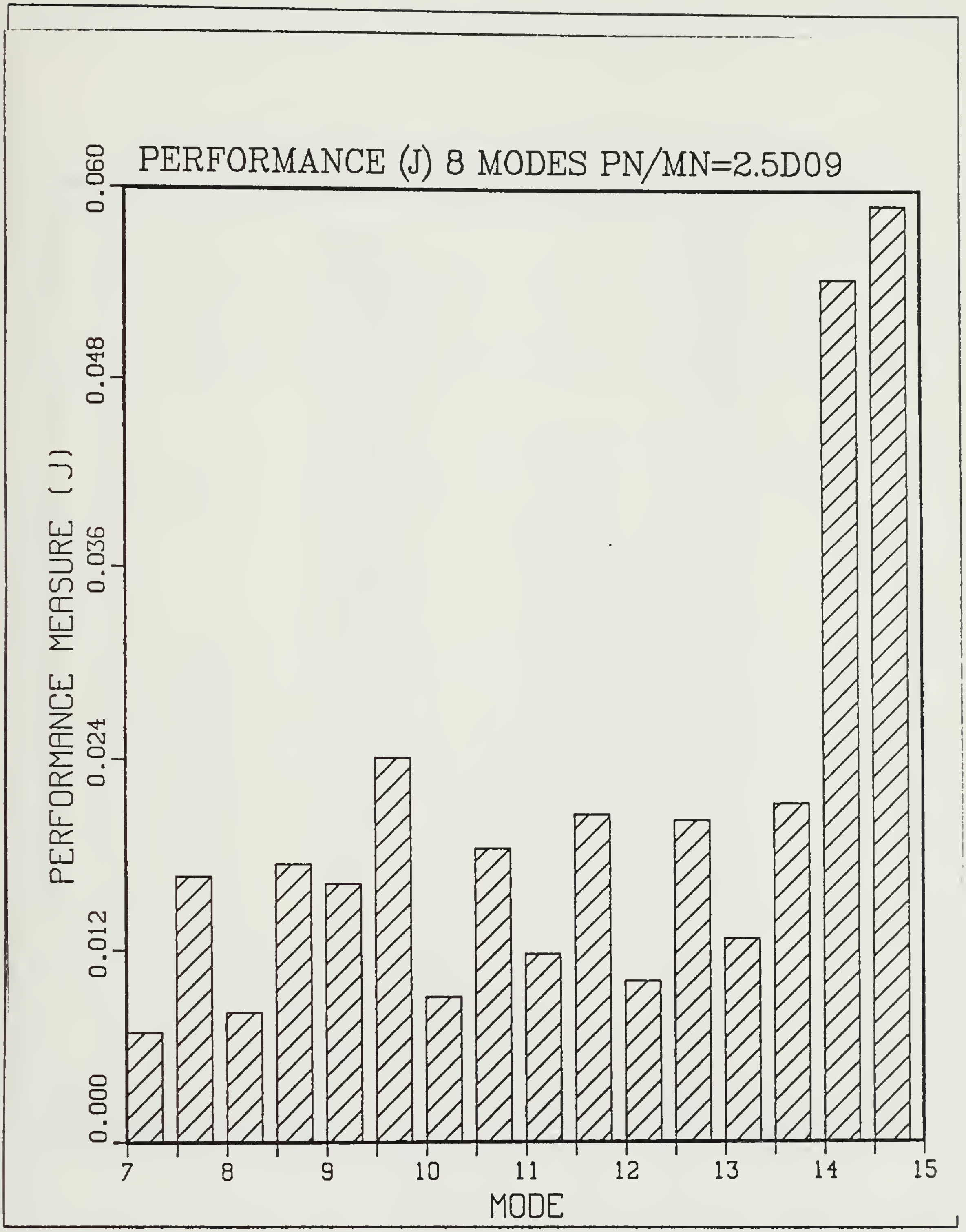

Figure 13. Observer Performance (J) 8 Modes (7 - 14)

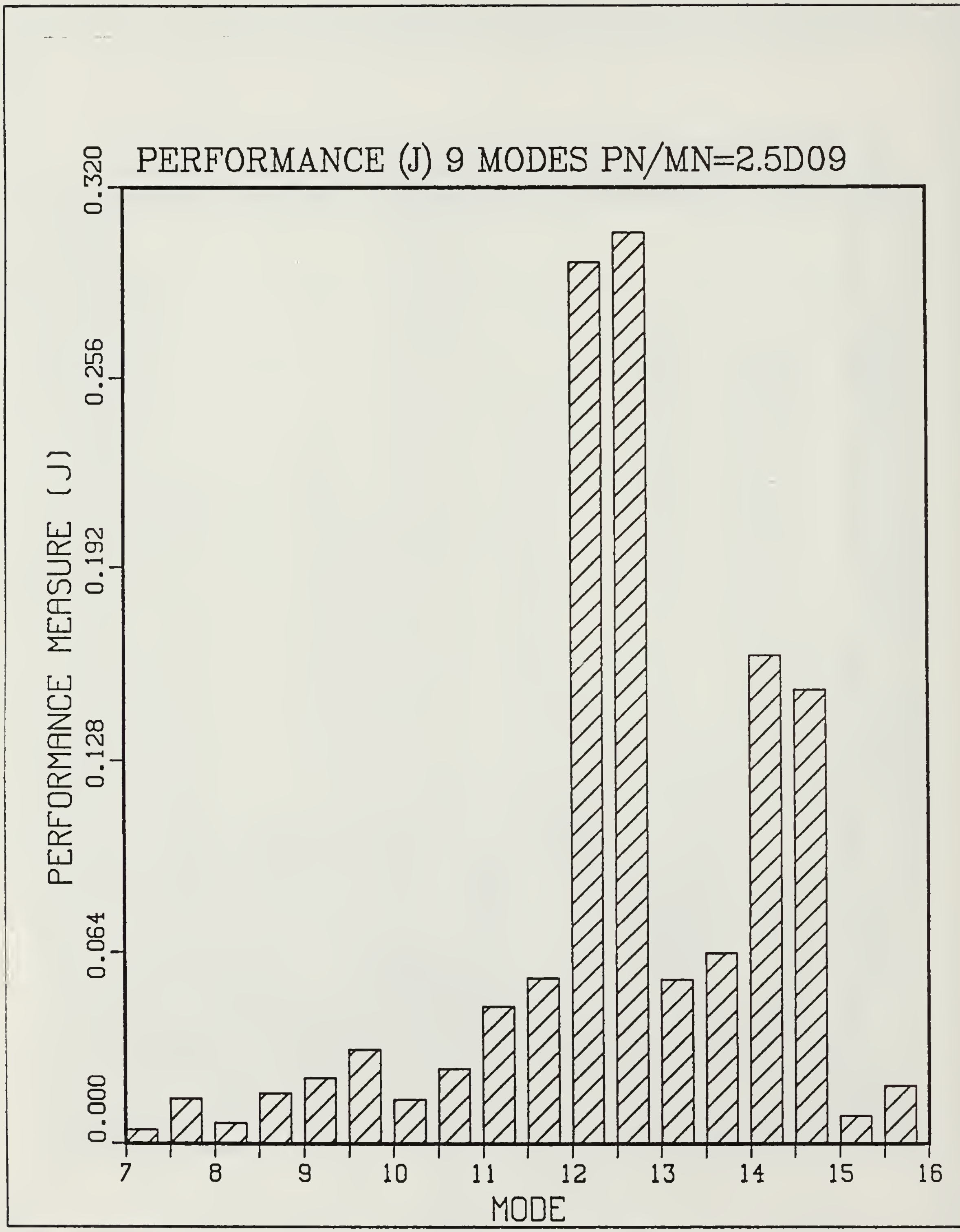

Figure 14. Observer Performance (J) 9 Modes (7 - 15)

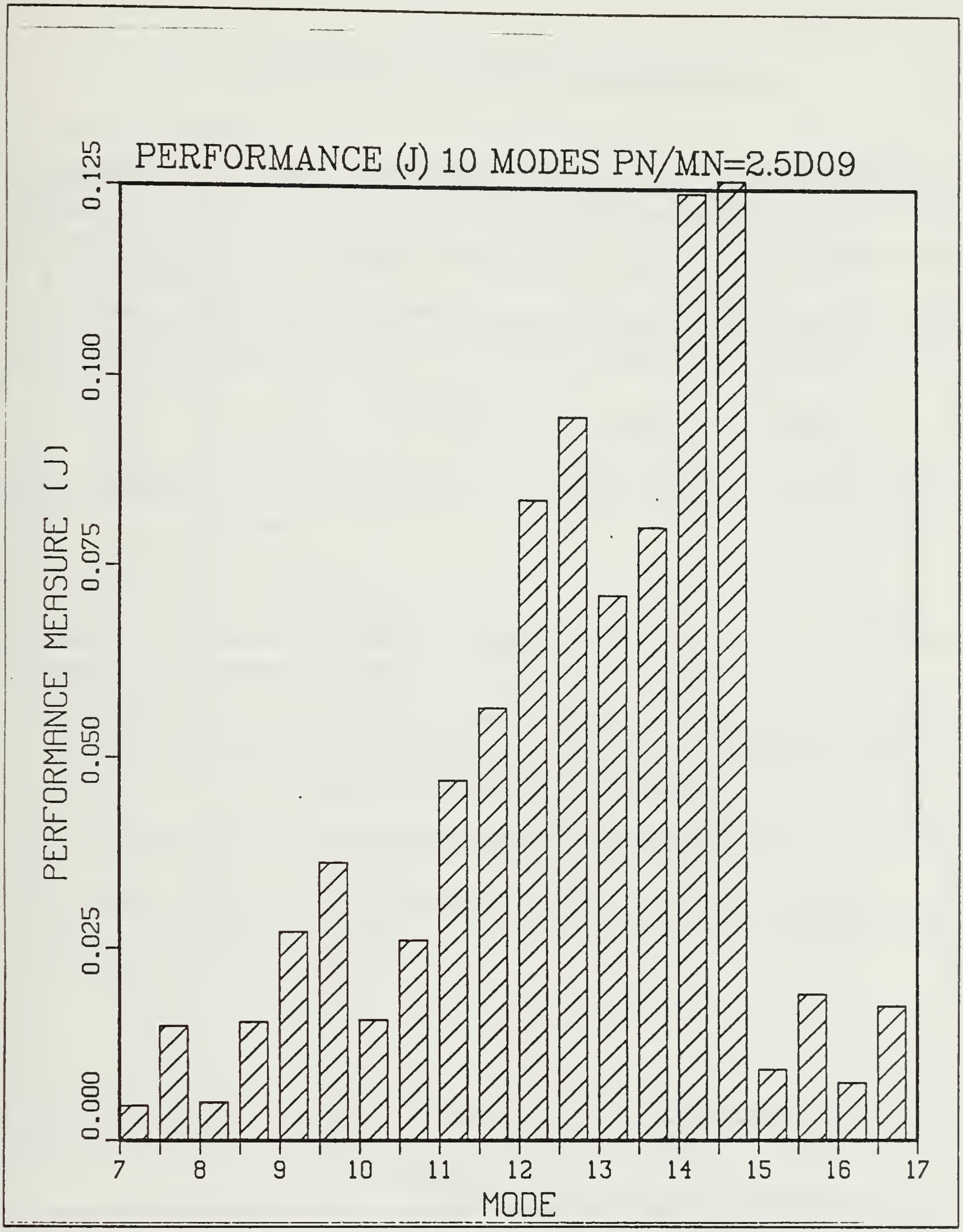

Figure 15. Observer Performance (J) 10 Modes (7 - 16)

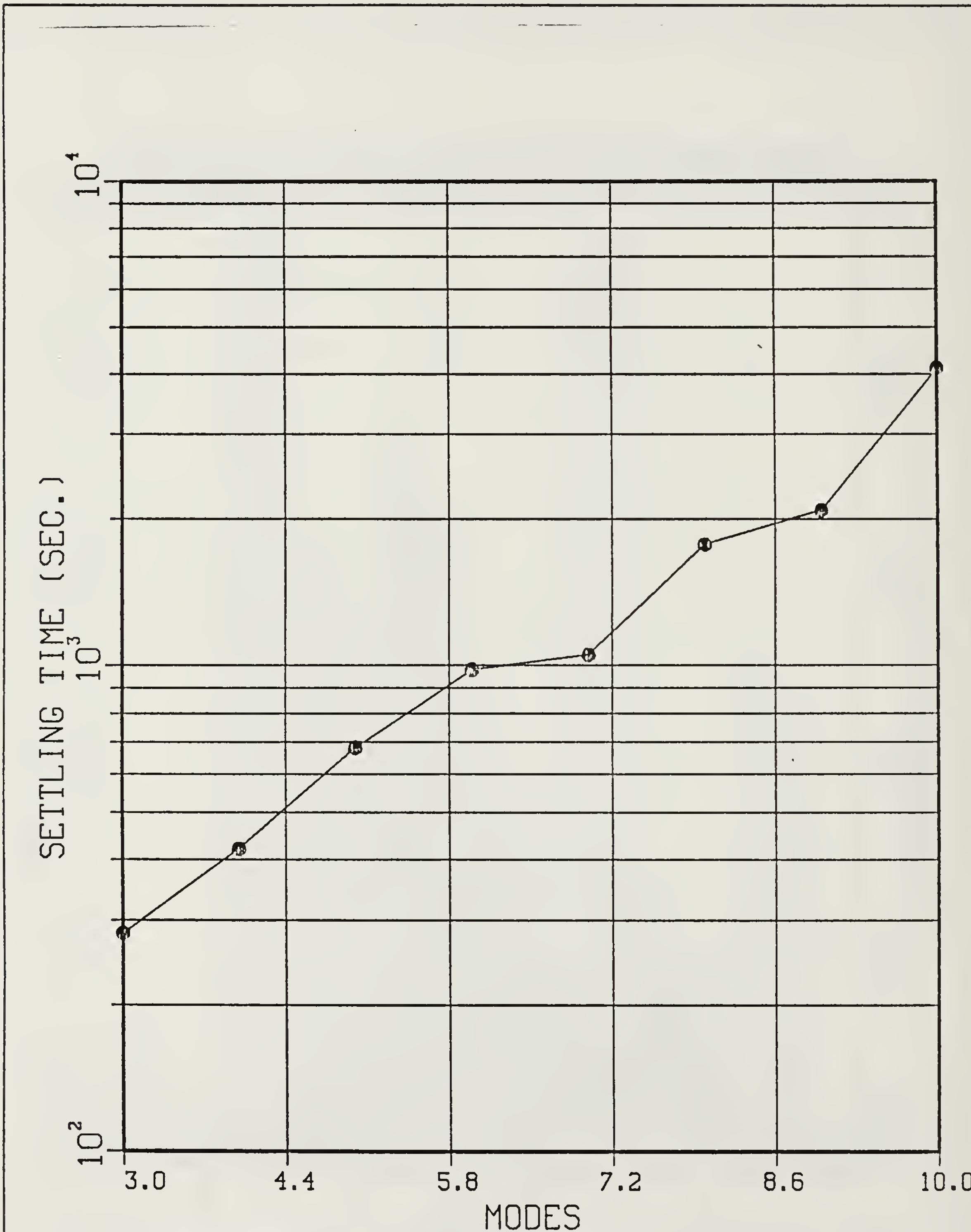

Figure 16. Settling Time versus number of Modes Observed

V. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

Simulations runs showed that a matched plant/observer can work if the following criterions are meet:

- The ratio of plant noise to measurement noise is sufficiently high to produce a usable settling time.
- Sufficient computational power is available to run the matched observer. The amount of memory and number of computation goes up as the number of modes observed increases.

Utilizing a reduce order observer for an arbitrarily selected set of modes is not feasible. The non-observed modes add so much noise to the system that settling times and observer performance are so poor as to render the observer useless for obtaining state values for plant control.

B. RECOMENDATIONS

The work on the Kalman Observer for Large Space Structures lead to the following recommendations for further research:

- Identify those modes that contribute the largest noise to the Kalman observer and set the observer to estimate these states in addition to those required for plant control. A possible method for identifying the modes that contibute the largest noise to the observer would be the Karhunen-Loeve expansion.
- Modifying the plant/observer to model the use of sensors at additional positions to see if the increase in the data rate will help decrease settling time.
- Modify the model to incorporate noise injection into more than one location . The current model has noise injected at only one position, a useful simplification for initial analysis but not realistic.

APPENDIX A. KALMAN GAIN MATRIX GENERATION PROGRAM

```
C
C     ************************************************************
C     *****               GGAIN                              *****
C     *****  ADAPTED TO RUN KALMAN FILTER AND COMPUTE THE    *****
C     *****  G MATRIX BY ITERATION STOPPING WHEN THE         *****
C     *****  THE MATRIX GOES TO STEADY STATE                 *****
C     ************************************************************
C
C
C     ************************************************************
C     *****            VARIABLE DECLARATIONS                 *****
C     ************************************************************
C
      EXTERNAL STMTRX,DLINRG,EXCMS, DEVCRG
      CHARACTER*6 NAM
      CHARACTER*1 AGAIN,CORECT,RAGAIN
      INTEGER ROWN1,ROWN2,ROWN3,COUNT,NODE,MODE,KQ,EMODE,SMODE,R2M,C2M
      INTEGER CT,CF,KADJ,CFADJ,LOOP,PRNT,JJ,JK,N1,JR,KR,MR,ISEED,M2
      INTEGER JL,J1,JM
      REAL LAMA(100), UGVEX(684,100),RNODE,RMODE,MIN
      REAL*8 PHI(2,2,100),GAMMA(2,100),EGT,GMA,WN,W1,X1T,X2T,TIME
      REAL*8 A(200,200),B(200,3),F(3, 50),IMPLSE,ENERGY
      REAL*8 COSW1T,SINW1T,X1(100),X2(100),COST(100)
      REAL*8 DAMP,SAMPT,PI,SAMPTM,SUM1,SUM2,SUM3,SUMC
      REAL*8 C(6,200), IDENT( 50, 50), RMN(6,6), QPN(3,3)
      REAL*8 PK( 50, 50), Y(6), BN(200,3)
      REAL*8 PNVARX, PNVARY, PNVARZ
      REAL*8 MNVX1, MNVY1, MNVZ1, SUM, BQBT(50,50)
      REAL*8 TMP1( 50,3), TMP2(3,3), TMP3( 50, 50)
      REAL*8 PK1( 50, 50),G( 50,3)
      REAL*8 DY(3), ES,ED,ESUM,CGN,PRT
      REAL*8 SF, N9, TCHK, ACHK, H1, H2, H3, H4, H5, H6
      REAL*8 AGC(100,100)
C
      COMPLEX*8 EVAL(100), EVEC (100,100)
C
C
C
C     ************************************************************
C     *****            VARIABLE DEFINITIONS                  *****
C     ************************************************************
C
C     STMTRX = SUBROUTINE EXTABLISHES STATE TRANSITION MATRICIES
C     LAMA = VECTOR OF THE SQUARE OF THE NATURAL FREQUENCIES
C     UGVEX = MODE POSITONS AND SLOPES OF THE NODAL POINTS
C     PHI = STATE TRANSITION MATRICIES FOR EACH MODE
C     GAMMA = INPUT TRANSITION MATRIX
C     A = DIAGONAL MATRIX CONSISTING OF PHI
C     B = INPUT MATRIX OF GAMMA AND CONTROL SLOPES
C     DAMP = DAMPING FACTOR
C     SAMPT = SAMPLING TIME
```

```
C       TCX, TCY, TCZ = CONTROL TORQUE VALUES
C       ENERGY = TOTAL SYSTEM ENERGY
C       IMPLSE = IMPULSE INPUT FUNCTION
C       MIN = NUMBER OF MINUTES SYSTEM WILL BE OBSERVED
C       SMODE = NUMBER OF STARTING MODE (INT)
C       MODE = NUMBER OF MODES (INT)
C       EMODE = NUMBER OF THE LAST MODE (INT)
C       NODE = NUMBER OF THE NOISE INPUT MODE (INT)
C        *** NOISE SLOPE LOCATIONS IN DATA MATRIX ***
C       ROWN1 = X-SLOPE LOCATION
C       ROWN2 = Y-SLOPE LOCATION
C       ROWN3 = Z-SLOPE LOCATION
C       C = OUTPUT MATRIX FOR Y
C       IDENT = IDENTITY MATRIX
C       RMN = MEASUREMENT NOISE COVARIANCE MATRIX
C       QPN = PLANT NOISE COVARIANCE MATRIX
C       PNVARX = PLANT NOISE X-SLOPE VARIANCE
C       PNVARY = PLANT NOISE Y-SLOPE VARIANCE
C       PNVARZ = PLANT NOISE Z-SLOPE VARIANCE
C       MNVARX = MEASUREMENT NOISE X-SLOPE VARIANCE
C       MNVARY = MEASUREMENT NOISE Y-SLOPE VARIANCE
C       MNVARZ = MEASUREMENT NOISE Z-SLOPE VARIANCE
C       ISEED = INITIALIZATION FOR RANDOM NUMBER GENERATOR
C       XKAL = X MATRIX
C       Y = OUTPUT MATRIX
C       RNDM = RANDOM NUMBERS USED FOR WHITE NOISE IN MEASUREMENTS AND
C               IN PLANT FORCES
C       BN = B MATRIX TO MULTIPLY NOISE DISTURBANCES
C       TNX,TNY,TNZ= NOISE TORQUES X,Y,Z SLOPES
C       M2=2*MODE
C
C
C
C       **********      SAMPLE OF SPACE EXEC FILE      ******************
C
C       THIS FILE MUST BEGIN IN COLUMN 1 AND RUN WITH THE FOLLOWING
C       SEQUENCE FOR THE INITIAL RUN OF THE PROGRAM:
C
C               FORTVS SPACE            (COMPILES PROGRAM)
C               SPACE                   (EXECUTES EXEC FILE)
C               LOAD SPACE (START       (LOADS AND EXECUTES PROGRAM)
C
C       SUBSEQUENT PROGRAM RUNS CAN ELIMINATE "FORTVS SPACE" IF NO
C       CHANGES HAVE BEEN MADE TO THE PROGRAM, AND CAN ELIMINATE
C       RUNNING THE EXEC FILE.
C
C       FI 4 DISK THESIS INPUT B (PERM
C       FI 8 DISK UTILITY DATA  (RECFM VS BLOCK 133 PERM
C       FI 11 DISK CNTRL OUTPUT  (RECFM F BLOCK 80 LRECL 80 PERM
C       FI 13 DISK GAMMA OUTPUT  (RECFM VS BLOCK 133 PERM
C       FI 14 DISK MODE OUTPUT  (RECFM F BLOCK 80 LRECL 80 PERM
C       FI 16 DISK COST OUTPUT  (RECFM F BLOCK 80 LRECL 80 PERM
C       FI 17 DISK PRT  OUTPUT  (RECFM F BLOCK 80 LRECL 80 PERM
C       FI 18 DISK ERROR DATA   (RECFM F BLOCK 80 LRECL 80 PERM
C       FI 19 DISK END FILE   (RECFM F BLOCK 80 LRECL 80 PERM
C       FI 20 DISK GMAT FILE (RECFM F BLOCK 80 LRECL 80 PERM
```

```
C
C     ************************************************************************
C
      PARAMETER (JR=5243, KR=5397, MR=262139)
C
      MIN =1200.0
      WT=1.0D00
      PI = 4.0D0 * ATAN(1.0D0)
C
C     *********************************************************************
C     *****           READ LAMA AND UGVEX MATRICIES                  *****
C     *********************************************************************
C
      CALL EXCMS ('CLRSCRN')
      WRITE(6,1008)
      WRITE(6,*) '                READING LAMA AND UGVEX MATRICIES'
      WRITE(6,*) ' '
C     THIS SECTION READS THE LAMA VECTOR AND THE UGVEX
C     MATRIX AND STORES THEM IN MEMORY FOR FURTHER RECALL OF
C     DESIRED LOCATION DATA.
C
      READ(4,1001) NAM
      READ(4,1002)(LAMA(I),I=1,100)
      READ(4,1001) NAM
      DO 5 J = 1,100
         READ(4,1002)(UGVEX(I,J),I=1,684)
5     CONTINUE
C
1001  FORMAT(1X,A6)
1002  FORMAT(1X,8E15.8)
1008  FORMAT(1X,////)
C
500   CALL EXCMS ('CLRSCRN')
C
C     *************          STARTING MODE NUMBER        **************
C     ** SMODE 7 TO 100 (INTEGER) ****
      SMODE=10
C
      WRITE (16,700) SMODE
700   FORMAT (' ','STARTING MODE NUMBER:  ',I2)
C
C     ***************       NUMBER OF MODES TO SCAN      *************
C     ** MODE 1 TO 93 (INTEGER)
C
      MODE= 3
C
      EMODE = SMODE + MODE - 1
C
      WRITE (16,701) MODE
701   FORMAT (' ','NUMBER OF MODES SCANNED:  ',I2)
C
C     *************       NOISE INPUT POSITION      ****************
C     ** NODE 1 TO 114 (INTEGER) (IF 0 THEN NO NOISE INPUT)
      NODE= 8
C
```

```
      WRITE (16,702) NODE
702   FORMAT (' ',' NOISE NODE LOCATION:    ',I5)
C
C
C     ***************          SAMPLING TIME           ***************
C     ** SAMPT MUST BE LESS THAN OR EQUAL TO SAMPTM **
      SAMPT = .05
         SAMPTM = ((2.0D0*PI)/SQRT(LAMA(EMODE)))/2.0D0
         IF (SAMPT.GE.SAMPTM) THEN
             SAMPT=SAMPTM
         ENDIF
C
      WRITE (16,900) MIN
900   FORMAT (' ',2X,'MIN:   ',F8.3)
C
      WRITE (16,703) SAMPT
703   FORMAT (' ','SAMPLING TIME:   ',D12.4)
C
C     *****************          DAMPING FACTOR          ***************
C     ** DAMP 0.0 TO 1.0 (REAL*8)
      DAMP=.01
C
      WRITE (16,704) DAMP
704   FORMAT (' ','DAMPING FACTOR:   ',D12.4)
C
C
C     *** PLANT NOISE VARIANCE ***
C     ** PNVARX, PNVARY, PNVARZ GT 0.0
C
      SF1=2.5D06
C
      PNVARX=1.0D00*SF1
      PNVARY=1.0D00*SF1
      PNVARZ=1.0D00*SF1
C
C
C
C     *** MEASUREMENT NOISE VARIANCE ***
C     ** MNVARX, MNVARY, MNVARZ GT 0.0
      SF=1.0
      MNVX1=5.5D-03*SF
      MNVY1=5.5D-03*SF
      MNVZ1=5.5D-03*SF
C
510   CALL EXCMS ('CLRSCRN')
      WRITE (6,1008)
      WRITE (6,*) '                               PROGRAM RUNNING'
C
C     ***********         NOISE INPUT LOCATION          *************
C
      ROWN3 = NODE*6
      ROWN2 = (NODE*6) - 1
      ROWN1 = (NODE*6) - 2
      COUNT = 0
C
```

```
C      ***********       INITIALIZE MATRICIES       ****************
C
       DO 40 I = 1,3
          DO 45 J = 1,3
              RMN(I,J)=0.0
45        CONTINUE
40     CONTINUE
C
       DO 47 I=1,50
          DO 46 J=1,50
          IDENT(I,J)=0.0
          PK(I,J)=0.0
46        CONTINUE
47     CONTINUE
C
       DO 48 K=1,50
          IDENT(K,K)=1.0
48     CONTINUE
C
C      *** INITIALIZE RMN AND QPN MATRICES ***
C
       DO 60 I=1,3
          DO 58 J=1,3
          QPN(I,J)=0.0
58        CONTINUE
60     CONTINUE
C
       RMN(1,1)=MNVX1**2
       RMN(2,2)=MNVY1**2
       RMN(3,3)=MNVZ1**2
       QPN(1,1)=PNVARX**2.0
       QPN(2,2)=PNVARY**2.0
       QPN(3,3)=PNVARZ**2.0
C
9999   FORMAT (' ','   ')
C      ***********************************************************
C      *****              BEGIN MAIN PROGRAM                *****
C      ***********************************************************
C
       CALL STMTRX(EMODE,SMODE,SAMPT,DAMP,PHI,GAMMA,A,B,LAMA,UGVEX,C,
      +   ROWN1,ROWN2,ROWN3,BN)
C
C
C      *** PRE-LOOP PORTION OF KALMAN FILTER
       JK=SMODE*2-2
       M2=2*MODE
       DO 94 I=1,3
          DO 92 J=1,M2
          JL=JK+J
          SUM=0.0
               DO 90 K=1,3
               SUM=SUM+QPN(I,K)*BN(JL,K)
90             CONTINUE
          TMP1(J,I)=SUM
92        CONTINUE
94     CONTINUE
```

```
C
C
      DO 98 I=1,M2
      JL=JK+I
         DO 97 J=1,M2
         SUM=0.0
              DO 96 K=1,3
              SUM=SUM+BN(JL,K)*TMP1(J,K)
96            CONTINUE
         BQBT(I,J)=SUM
97       CONTINUE
98    CONTINUE
C
      M2=2*MODE
      DO 100 I=1,M2
         DO 99 J=1,M2
         TMP3(I,J)=0.0
99       CONTINUE
100   CONTINUE
      JL=JK+M2
      DO 9375 I=1,3
         DO 9374 J=1,JL
         C(I,J)=C(I,J)*SF
9374     CONTINUE
9375  CONTINUE
C
C     **********************************************************
C     *****     THIS SECTION COMPUTES THE STATE UPDATE     *****
C     **********************************************************
      ESUM=0.0
      COUNT = 0
      ENERGY = 0.0D0
      TIME = 0.0
      CGN=0.0
C     *****  SETS LOOP FOR THE ITERATIONS NECESSARY TO OBSERVE  *****
C     *****  THE SYSTEM FOR THE NUMBER OF MINUTES SPECIFIED     *****
C
      LOOP = INT((MIN*60.0)/SAMPT)
      PRT=(DBLE(LOOP))/1200.0
      PRTA=(DBLE(LOOP))/2400.0
      CNTA=0.0
      ACHK=1.0D-10
      H1=0.0
      H2=0.0
      H3=0.0
      H4=0.0
      H5=0.0
      H6=0.0
      TCHK=MIN*60.0
9991  CONTINUE
C
         TIME = TIME+   SAMPT
C
      CGN=CGN+1.0
C     CNTA=CNTA+1.0
```

```
C     *** START OF KALMAN FILTER ***
C
      M2=2*MODE
C
C     *** COMPUTATION OF PK*AT ***
C
      JK=2*SMODE-2
      DO 175 I=1,M2
         DO 170 J=1,M2
         JL=JK+J
         SUM=0.0
              DO 165 K=1,M2
              JM=JK+K
              SUM =SUM+PK(I,K)*A(JL,JM)
165           CONTINUE
         TMP3(I,J)=SUM
170      CONTINUE
175   CONTINUE
C
C
C     *** COMPUTATION OF A*(PK*AT)+ BQBT = PK1 ***
C
      DO 190 I=1,M2
      JL=JK+I
         DO 185 J=1,M2
         SUM=0.0
              DO 180 K=1,M2
              JM=JK+K
              SUM=SUM+A(JL,JM)*TMP3(K,J)
180           CONTINUE
         PK1(I,J)=SUM+BQBT(I,J)
185      CONTINUE
190   CONTINUE
C     ***********
C
C     *** COMPUTE PK1*CT ****
C
      DO 205 I=1,M2
         DO 200 J=1,3
         SUM=0.0
              DO 195 K=1,M2
              JM=JK+K
              SUM=SUM+PK1(I,K)*C(J,JM)
195           CONTINUE
         TMP1(I,J)=SUM
200      CONTINUE
205   CONTINUE
C     ************
C
C     *** COMPUTE C*(PK1*CT)+RMN ***
      DO 220 I=1,3
         DO 215 J=1,3
         SUM=0.0
              DO 210 K=1,M2
              JM=JK+K
```

```
            SUM=SUM+C(I,JM)*TMP1(K,J)
210         CONTINUE
        TMP2(I,J)=SUM+RMN(I,J)
215     CONTINUE
220   CONTINUE
C
C     *** COMPUTATION OF THE INVERSE OF C*PK1*CT + R
C
C
      CALL DLINRG ( 3,TMP2,3,TMP2,3)
C
C     *** COMPUTE CT*INV(C*PK1*CT+R)
C
      DO 245 I=1,M2
      JL=JK+I
        DO 240 J=1,3
        SUM=0.0
            DO 235 K=1,3
            SUM=SUM+C(K,JL)*TMP2(K,J)
235         CONTINUE
        TMP1(I,J)=SUM
240     CONTINUE
245   CONTINUE
C     ***************
C
C     *** COMPUTE PK1*C*INV(C*PK1*CT+R) = G *****
C
      DO 260 I=1,M2
        DO 255 J=1,3
        SUM=0.0
            DO 250 K=1,M2
            SUM=SUM+PK1(I,K)*TMP1(K,J)
250         CONTINUE
        G(I,J)=SUM
255     CONTINUE
260   CONTINUE
C
      N9=DABS((G(1,1)-H1)/G(1,1))
      IF (N9.GT.ACHK)     THEN
      GO TO 7377
      END IF
      N9=DABS((G(1,3)-H2)/G(1,3))
      IF (N9.GT.ACHK)THEN
      GO TO 7377
      END IF
      N9=DABS((G(2,1)-H3)/G(2,1))
      IF (N9.GT.ACHK)     THEN
      GO TO 7377
      END IF
      N9=DABS((G(2,3)-H4)/G(2,3))
      IF (N9.GT.ACHK)     THEN
      GO TO 7377
      END IF
      N9=DABS((G(3,3)-H5)/G(3,3))
      IF (N9.GT.ACHK)     THEN
      GO TO 7377
```

```
      END IF
      N9=DABS((G(M2,3)-H6)/G(M2,3))
      IF (N9.GT.ACHK)     THEN
      GO TO 7377
      END IF
      GO TO 400
C
C
7377  CONTINUE
      H1=G(1,1)
      H2=G(1,3)
      H3=G(2,1)
      H4=G(2,3)
      H5=G(3,3)
      H6=G(M2,3)
      IF (TCHK.LE.TIME) THEN
      GO TO 400
      END IF
      IF (CGN.GE.PRT)   THEN
C
      WRITE (6,*) 'TIME= ', TIME, ' SEC.'
C
      WRITE (6,*) 'N9= ', N9
      CGN=0.0
      END IF
C
C     *** COMPUTE IDENT  - G*C
C
      DO 275 I=1,M2
         DO 270 J=1,M2
         JL=JK+J
         SUM=0.0
              DO 265 K=1,3
              SUM=SUM+G(I,K)*C(K,JL)
265           CONTINUE
         TMP3(I,J)= IDENT(I,J)-SUM
270      CONTINUE
275   CONTINUE
C     **********************
C
C     *** COMPUTE PK= (INDENT - G*C)*PK1
C
      DO 290 I=1,M2
         DO 285 J=1,M2
         SUM=0.0
              DO 280 K=1,M2
              SUM=SUM+TMP3(I,K)*PK1(K,J)
280           CONTINUE
         PK(I,J)=SUM
285      CONTINUE
290   CONTINUE
C     **********************
C
```

```
C     END OF KALMAN FILTER PART 1 - START OF PART 2 ****
C
C
C
      GO TO 9991
C
400   CONTINUE
C
      WRITE (20,1008)
      WRITE (20,*) 'TIME= ',TIME
      DO 384 I=1,M2
      WRITE (20,5350)   G(I,1),G(I,2),G(I,3)
384   CONTINUE
5350  FORMAT (' ',5X,D15.8 ,5X,D15.8 ,5X,D15.8 )
      WRITE (20,*) 'N9= ',N9
C
C     *** COMPUTE AGC = A - G*C
C
      M2=2*MODE
      JK=2*SMODE-2
C
      DO 7155 I=1,M2
      JL=JK+I
         DO 7154 J=1,M2
         JM=JK+J
         SUM=0.0
              DO 7153 K=1,3
              SUM=SUM+G(I,K)*C(K,JM)
7153          CONTINUE
         AGC(I,J)=A(JL,JM)-SUM
7154     CONTINUE
7155  CONTINUE
C
C
C
C     *** COMPUTE THE EIGENVALUES OF AGC
C
      CALL DEVCRG (M2, AGC, 100, EVAL, EVEC, 100)
C
C     **** PRINT EVAL (EIGENVALUE) MATRIX
C
      DO 7157 I=1,M2
      WRITE (20,*) 'I= ', I, 'EIG= ', EVAL(I)
7157  CONTINUE
C
C
C
599   STOP
      END
C
C
C
C
C
C     ***********************************************************************
C     THIS SUBROUTINE COMPUTES THE STATE TRANSITION MATRIX FOR EACH
```

```
C     OF THE 100 MODES
C     ***************************************************************
C
      SUBROUTINE STMTRX(EMODE,SMODE,T,D,PHI,GAMMA,A,B,LAMA,UGVEX,C,
     +        ROWN1,ROWN2,ROWN3,BN)
C
      REAL*8 WN,GMA,PHI(2,2,100),GAMMA(2,100),EGT,T,COSW1T,SINW1T
      REAL*8 W1,D,A(200,200),B(200,3),C(6,200),BN(200,3)
      REAL LAMA(100),UGVEX(684,100)           .
      INTEGER SMODE,R,EMODE,JJ,KK,ROWN1,ROWN2,ROWN3
C
C
C
C
C
      DO 600 I = 1    ,100
         WN = DBLE(SQRT(LAMA(I)))
         GMA = D*WN/2.0
         EGT = DEXP(-GMA*T)
         W1 = DSQRT((WN**2)-(GMA**2))
         COSW1T = DCOS(W1*T)
         SINW1T = DSIN(W1*T)
C
C
C
C
C
         IF(WN.EQ.0)THEN
              PHI(1,1,I) = EGT*COSW1T
              PHI(1,2,I) = T
              PHI(2,1,I) = 0
              PHI(2,2,I) = EGT*COSW1T
C
C
C
C
C
C
              GAMMA(1,I) = 0
              GAMMA(2,I) = 0
         ELSE
C
C
C
C
C
         PHI(1,1,I) = EGT*(COSW1T + (GMA*(W1**(-1)))*SINW1T)
         PHI(1,2,I) = (W1**(-1))*EGT*SINW1T
         PHI(2,1,I) = -(WN**2)*(W1**(-1))*EGT*SINW1T
         PHI(2,2,I) = EGT*(COSW1T - (GMA*(W1**(-1)))*SINW1T)
C
C
C
C
      GAMMA(1,I)=(WN**(-2))*(1.D0-EGT*COSW1T-EGT*(GMA/W1)*SINW1T)
```

```
         GAMMA(2,I) = (W1**(-1))*EGT*SINW1T
C
C
C
C
         ENDIF
C
C
C
600   CONTINUE
C
C
C
C
      R = 1
C
      DO 610 K = 1    ,100
C
C
C
C
C
         A(R,R) = PHI(1,1,K)
         A(R,R+1) = PHI(1,2,K)
         A(R+1,R) = PHI(2,1,K)
         A(R+1,R+1) = PHI(2,2,K)
C
C
C
C
C
C     *** B MATRIX FOR MULTIPLYING CONTROL TORQUES
C
         B(R,1) = GAMMA(1,K)*DBLE(UGVEX(412,K))
         B(R,2) = GAMMA(1,K)*DBLE(UGVEX(413,K))
         B(R,3) = GAMMA(1,K)*DBLE(UGVEX(414,K))
         B(R+1,1) = GAMMA(2,K)*DBLE(UGVEX(412,K))
         B(R+1,2) = GAMMA(2,K)*DBLE(UGVEX(413,K))
         B(R+1,3) = GAMMA(2,K)*DBLE(UGVEX(414,K))
C
C
C
C
C
C
C
C     *** BN MATRIX FOR MULTIPLYING THE NOISE DISTURBANCES
C
C
C
C
      BN(R,1)=GAMMA(1,K)*DBLE(UGVEX(ROWN1,K))
      BN(R,2)=GAMMA(1,K)*DBLE(UGVEX(ROWN2,K))
      BN(R,3)=GAMMA(1,K)*DBLE(UGVEX(ROWN3,K))
      BN(R+1,1)=GAMMA(2,K)*DBLE(UGVEX(ROWN1,K))
      BN(R+1,2)=GAMMA(2,K)*DBLE(UGVEX(ROWN2,K))
```

```
      BN(R+1,3)=GAMMA(2,K)*DBLE(UGVEX(ROWN3,K))
C
C
C
C
C
C
         R = R+2
610   CONTINUE
C
C
C
C
C
C
C
C     *** C MATRIX PRODUCTION ***
C
C
C
      JJ=-1
      DO 640 I=1,100
      JJ=JJ+1
      KK=I+JJ
C
C
      C(1,KK) = DBLE(UGVEX(418,I))
      C(2,KK) = DBLE(UGVEX(419,I))
      C(3,KK) = DBLE(UGVEX(420,I))
C
C
C
      KK=KK+1
C
      C(1,KK)=0.0
      C(2,KK)=0.0
      C(3,KK)=0.0
C
640   CONTINUE
C
C
C
      RETURN
      END
```

APPENDIX B. KALMAN OBSERVER AND PLANT SIMULATION

```
C
C     ***********************************************************
C     *****                SIMRUN                           *****
C     *****  ADAPTED TO READ KALMAN FILETER G MATRICE        *****
C     *****  THEN RUN ALL N   MODES OF THE PLANT WHILE      *****
C     *****  USING A KALMAN FILTER TO OBSERVE M             *****
C     *****  NUMBER OF STATES                               *****
C     ***********************************************************
C
C
C     ***********************************************************
C     *****            VARIABLE DECLARATIONS                *****
C     ***********************************************************
C
      EXTERNAL STMTRX,EXCMS
      CHARACTER*6 NAM
      CHARACTER*1 AGAIN,CORECT,RAGAIN
      INTEGER ROWN1,ROWN2,ROWN3,COUNT,NODE,MODE,KQ,EMODE,SMODE,R2M,C2M
      INTEGER CT,CF,KADJ,CFADJ,LOOP,PRNT,JJ,JK,N1,JR,KR,MR,ISEED,M2
      INTEGER ITYPE(200), IPVT(100), NS, NF, SN, FN
      INTEGER JL,J1,JM , JP, JQ, KA, KB, KC, KD, KE, KF, KG
C
C
C
      REAL LAMA(100), UGVEX(684,100),RNODE,RMODE,MIN
      REAL*8 PHI(2,2,100),GAMMA(2,100),EGT,GMA,WN,W1,X1T,X2T,TIME
      REAL*8 A(200,200),B(200,3),F(3, 50),IMPLSE,ENERGY
      REAL*8 COSW1T,SINW1T,X(200)
      REAL*8 TCX,TCY,TCZ,DAMP,SAMPT,PI,SAMPTM,SUM1,SUM2,SUM3,SUMC
      REAL*8 C(3,200), IDENT( 50, 50), RMN(3,3), QPN(3,3)
      REAL*8 Y(3)  , BN(200,3)
      REAL*8 PNVARX, PNVARY, PNVARZ
      REAL*8 MNVARX, MNVARY, MNVARZ
      REAL*8 SUM, RNDM(6), RND1, RND2
      REAL*8 XH( 50)  ,BQBT( 50, 50)
      REAL*8 SF1
      REAL*8 TMP1( 50,3), TMP2(3,3), TMP3( 50, 50)
      REAL*8 G( 50,3)
      REAL*8 XH1( 50)  ,DY(3)  , ES,ED,ESUM,CGN,PRT
      REAL*8 WT , WD(3), BNWD(200)
      REAL*8 AX(200) , V(3), SF , TO, CTT, ESS
      REAL*8 CTG, XDEL, E2(100), XDEL1, ERS, PRT1, E3(100), XS(100)
C
C     ***********************************************************
C     *****            VARIABLE DEFINITIONS                 *****
C     ***********************************************************
C
C     STMTRX = SUBROUTINE EXTABLISHES STATE TRANSITION MATRICIES
C     LAMA = VECTOR OF THE SQUARE OF THE NATURAL FREQUENCIES
C     UGVEX = MODE POSITONS AND SLOPES OF THE NODAL POINTS
C     PHI = STATE TRANSITION MATRICIES FOR EACH MODE
```

```
C     GAMMA = INPUT TRANSITION MATRIX
C     A = DIAGONAL MATRIX CONSISTING OF PHI
C     B = INPUT MATRIX OF GAMMA AND CONTROL SLOPES
C     DAMP = DAMPING FACTOR
C     SAMPT = SAMPLING TIME
C     TCX, TCY, TCZ = CONTROL TORQUE VALUES
C     ENERGY = TOTAL SYSTEM ENERGY
C     IMPLSE = IMPULSE INPUT FUNCTION
C     MIN = NUMBER OF MINUTES SYSTEM WILL BE OBSERVED
C     SMODE = NUMBER OF STARTING MODE (INT)
C     MODE = NUMBER OF MODES (INT)
C     EMODE = NUMBER OF THE LAST MODE (INT)
C     NODE = NUMBER OF THE NOISE INPUT MODE (INT)
C      *** NOISE SLOPE LOCATIONS IN DATA MATRIX ***
C     ROWN1 = X-SLOPE LOCATION
C     ROWN2 = Y-SLOPE LOCATION
C     ROWN3 = Z-SLOPE LOCATION
C     C = OUTPUT MATRIX FOR Y
C     IDENT = IDENTITY MATRIX
C     RMN = MEASUREMENT NOISE COVARIANCE MATRIX
C     QPN = PLANT NOISE COVARIANCE MATRIX
C     PNVARX = PLANT NOISE X-SLOPE VARIANCE
C     PNVARY = PLANT NOISE Y-SLOPE VARIANCE
C     PNVARZ = PLANT NOISE Z-SLOPE VARIANCE
C     MNVARX = MEASUREMENT NOISE X-SLOPE VARIANCE
C     MNVARY = MEASUREMENT NOISE Y-SLOPE VARIANCE
C     MNVARZ = MEASUREMENT NOISE Z-SLOPE VARIANCE
C     ISEED = INITIALIZATION FOR RANDOM NUMBER GENERATOR
C     XKAL = X MATRIX
C     Y = OUTPUT MATRIX
C     RNDM = RANDOM NUMBERS USED FOR WHITE NOISE IN MEASUREMENTS AND
C             IN PLANT FORCES
C     BN = B MATRIX TO MULTIPLY NOISE DISTURBANCES
C     TNX,TNY,TNZ= NOISE TORQUES X,Y,Z SLOPES
C     M2=2*MODE
C
C
C
C     **********     SAMPLE OF SPACE EXEC FILE     ********************
C
C     THIS FILE MUST BEGIN IN COLUMN 1 AND RUN WITH THE FOLLOWING
C     SEQUENCE FOR THE INITIAL RUN OF THE PROGRAM:
C
C             FORTVS SPACE           (COMPILES PROGRAM)
C             SPACE                  (EXECUTES EXEC FILE)
C             LOAD SPACE (START      (LOADS AND EXECUTES PROGRAM)
C
C     SUBSEQUENT PROGRAM RUNS CAN ELIMINATE "FORTVS SPACE" IF NO
C     CHANGES HAVE BEEN MADE TO THE PROGRAM, AND CAN ELIMINATE
C     RUNNING THE EXEC FILE.
C
C     FI 4 DISK THESIS INPUT B (PERM
C     FI 8 DISK UTILITY DATA  (RECFM VS BLOCK 133 PERM
C     FI 11 DISK CNTRL OUTPUT  (RECFM F BLOCK 80 LRECL 80 PERM
C     FI 13 DISK GAMMA OUTPUT  (RECFM VS BLOCK 133 PERM
C     FI 14 DISK MODE OUTPUT  (RECFM F BLOCK 80 LRECL 80 PERM
```

```
C     FI 16 DISK COST OUTPUT  (RECFM F BLOCK 80 LRECL 80 PERM
C     FI 17 DISK PRT  OUTPUT  (RECFM F BLOCK 80 LRECL 80 PERM
C     FI 18 DISK ERROR DATA   (RECFM F BLOCK 80 LRECL 80 PERM
C     FI 19 DISK END FILE   (RECFM F BLOCK 80 LRECL 80 PERM
C     FI 20 DISK GMAT FILE (RECFM F BLOCK 80 LRECL 80 PERM
C
C     ***************************************************************
C
      PARAMETER (JR=5243, KR=5397, MR=262139)
C
C
      MIN =1.00
C
      WT=1.0D00
      PI = 4.0D0 * ATAN(1.0D0)
C
C     **********************************************************
C     *****          READ LAMA AND UGVEX MATRICIES          *****
C     **********************************************************
C
      CALL EXCMS ('CLRSCRN')
      WRITE(6,1008)
      WRITE(6,*) '                     READING LAMA AND UGVEX MATRICIES'
      WRITE(6,*) ' '
C     THIS SECTION READS THE LAMA VECTOR AND THE UGVEX
C     MATRIX AND STORES THEM IN MEMORY FOR FURTHER RECALL OF
C     DESIRED LOCATION DATA.
C
      READ(4,1001) NAM
      READ(4,1002)(LAMA(I),I=1,100)
      READ(4,1001) NAM
      DO 5 J = 1,100
         READ(4,1002)(UGVEX(I,J),I=1,684)
5     CONTINUE
C
1001  FORMAT(1X,A6)
1002  FORMAT(1X,8E15.8)
1008  FORMAT (1X,////)
C
500   CALL EXCMS ('CLRSCRN')
C
C     ************       STARTING MODE NUMBER       **************
C     ** SMODE 7 TO 100 (INTEGER) ****
      SMODE= 7
C
      WRITE (16,700) SMODE
700   FORMAT (' ','STARTING MODE NUMBER:  ',I2)
C
C     ***************    NUMBER OF MODES TO SCAN    *************
C     ** MODE 1 TO 93 (INTEGER)
C
      MODE=20
C
      EMODE = SMODE + MODE - 1
C
```

```
      WRITE (16,701) MODE
701   FORMAT (' ','NUMBER OF MODES SCANNED:  ',I2)
C
C     **************    NOISE INPUT POSITION      *******************
C     ** NODE 1 TO 114 (INTEGER) (IF 0 THEN NO NOISE INPUT)
      NODE= 8
C
      WRITE (16,702) NODE
702   FORMAT (' ',' NOISE NODE LOCATION:   ',I5)
C
C     ************ START AND STOP FOR PLANT
      SN=7
      FN=20
      NS=SN*2-1
      NF=SN*2+2*FN-2
      WRITE (16,899) SN,FN
899   FORMAT (' ','PLANT -- SN= ',I5,'  FN= ',I5)
C     ***************          SAMPLING TIME          ***************
C     ** SAMPT MUST BE LESS THAN OR EQUAL TO SAMPTM **
      SAMPT = 0.05
         SAMPTM = ((2.0D0*PI)/SQRT(LAMA(EMODE)))/1.0D01
         IF (SAMPT.GE.SAMPTM) THEN
              SAMPT=SAMPTM
         ENDIF
C
      WRITE (16,900) MIN
900   FORMAT (' ',2X,'MIN:  ',F8.3)
C
      WRITE (16,703) SAMPT, SAMPTM
703   FORMAT (' ','SAMPLING TIME:  ',D12.4,2X,'SAMPTM= ',D15.8)
C
C     *****************         DAMPING FACTOR         ***************
C     ** DAMP 0.0 TO 1.0 (REAL*8)
      DAMP=.01
C
      WRITE (16,704) DAMP
704   FORMAT (' ','DAMPING FACTOR:  ',D12.4)
C
C
C     *** PLANT NOISE VARIANCE ***
C     ** PNVARX, PNVARY, PNVARZ GT 0.0
      SF1=2.5D06
      SF=1.0D00
C
      PNVARX=1.0D00*SF1
      PNVARY=1.0D00*SF1
      PNVARZ=1.0D00*SF1
C
C
C
C     *** MEASUREMENT NOISE VARIANCE ***
C     ** MNVARX, MNVARY, MNVARZ GT 0.0
      MNVARX=1.0D-03 *SF
      MNVARY=1.0D-03 *SF
```

```
      MNVARZ=1.0D-03 *SF
C
C
      WRITE (16,711)
711   FORMAT(' ','PLANT NOISE VARIANCE:  ')
      WRITE (16,712)
712   FORMAT(' ',6X,'PNVARX',13X,'PNVARY',13X,'PNVARZ')
      WRITE (16,713) PNVARX, PNVARY, PNVARZ
713   FORMAT(' ',2X,E15.8,2X,E15.8,2X,E15.8)
      WRITE(16,714)
714   FORMAT(' ','MEASUREMENT NOISE:')
      WRITE(16,715)
715   FORMAT(' ',6X,'MNVARX',13X,'MNVARY',13X,'MNVARZ')
      WRITE(16,713) MNVARX,MNVARY,MNVARZ
C
510   CALL EXCMS ('CLRSCRN')
      WRITE (6,1008)
      WRITE (6,*) '                              PROGRAM RUNNING'
C
C     ************         NOISE INPUT LOCATION         *************
C
      ROWN3 = NODE*6
      ROWN2 = (NODE*6) - 1
      ROWN1 = (NODE*6) - 2
      COUNT = 0
C
C
C     ************        INITIALIZE MATRICIES       *****************
C
      DO 48 K=1,50
         IDENT(K,K)=1.0
48    CONTINUE
C
      DO 54 K = 1, 200
          X(K) = 0.0
54    CONTINUE
C
C
      WRITE(6,1008)
      WRITE (6,*) ' INITIALIZE RMN AND QPN MATRICES   '
C     *** INITIALIZE RMN AND QPN MATRICES ***
C
      DO 60 I=1,3
         DO 58 J=1,3
         RMN(I,J)=0.0
         QPN(I,J)=0.0
58       CONTINUE
60    CONTINUE
C
      RMN(1,1)=MNVARX**2
      RMN(2,2)=MNVARY**2
      RMN(3,3)=MNVARZ**2
      QPN(1,1)=PNVARX**2
      QPN(2,2)=PNVARY**2
```

```
      QPN(3,3)=PNVARZ**2.0
C
C
      WRITE(6,1008)
      WRITE(6,*)  ' ENTER STMTRX    '
C
C     ***********************************************************
C     *****               BEGIN MAIN PROGRAM                *****
C     ***********************************************************
C
      CALL STMTRX(EMODE,SMODE,SAMPT,DAMP,PHI,GAMMA,A,B,LAMA,UGVEX,C,
     +   ROWN1,ROWN2,ROWN3,BN)
C
C
      WRITE (16,1008)
      DO 11000 I=1,200
         DO 10900 J=1,3
         C(J,I)= C( J,I)*SF
10900    CONTINUE
11000 CONTINUE
C
C     *** PRE-LOOP PORTION OF KALMAN FILTER
C
      M2=2*MODE
      JP=2*SMODE-1
      JQ=2*EMODE
      DO 90 I=1,50
         XH(I)=0.0
90    CONTINUE
C
      DO 9971 I=1,M2
      .  READ (20,*) G(I,1), G(I,2), G(I,3)
9971  CONTINUE
C
C
      WRITE (14,1008)
      DO 384 I=1,M2
      WRITE (14,5350)   G(I,1),G(I,2),G(I,3)
384   CONTINUE
5350  FORMAT (' ',2X,D15.8,2X,D15.8,2X,D15.8)
C
C
C     **********************************************************
C     *****     THIS SECTION COMPUTES THE STATE UPDATE     *****
C     **********************************************************
      DO 9771 I=1,100
      E2(I)=0.0
      E3(I)=0.0
      XS(I)=0.0
9771  CONTINUE
      ESS =0.0
      COUNT = 0
      ENERGY = 0.0D0
      TIME = 0.0
      CGN=0.0
```

```
      CTG=0.0
C     ***** SETS LOOP FOR THE ITERATIONS NECESSARY TO OBSERVE  *****
C     ***** THE SYSTEM FOR THE NUMBER OF MINUTES SPECIFIED     *****
      WRITE (6,1008)
      WRITE (6,*) '          START STATE UPDATE    '
      LOOP = INT((MIN*60.0)/SAMPT)
      PRT= (DBLE(LOOP))/30.0
      CTT=0.0
C
      DO 400 L  = 0, LOOP
         TIME = DBLE(L)*SAMPT
C
         IF(L.EQ.0)THEN
            IMPLSE =(1.0D06*SF1)/(DSQRT(SAMPT))
         ELSE
            IMPLSE = 0.0D0
         ENDIF
C
      TO=0.0
C     ***** RANDOM NUMBER GENERATOR *****
C
      DO 101 I=1,6
      ISEED=MOD(ISEED*JR+KR,MR)
      RND1=(DBLE(ISEED)+0.5D00)/DBLE(MR)
      ISEED=MOD(ISEED*JR+KR,MR)
      RND2=(DBLE(ISEED)+0.5D00)/DBLE(MR)
      RNDM(I)=DSQRT(-2.0*DLOG(RND1))*DCOS(6.2831853D00*RND2)
101   CONTINUE
C     *********************
      CTT=CTT+1.0
C     **** START OF STATE UPDATE ***
C
C     *** COMPUTE AX°200  = A°200 X 200  * X°200
C
C
C     *** COMPUTE AX  = A*X
C
      JK=SMODE*2-2
      JP=JK+1
      JQ=2*EMODE
C
C
      DO 5015 I=NS,NF
         SUM=0.0
             DO 5010 K=NS,NF
             SUM=SUM+A(I,K)*X(K)
5010         CONTINUE
         AX(I)=SUM
5015  CONTINUE
C
C     *** COMPUTE WD°3
C
      WD(1)=PNVARX*RNDM(1)*TO+IMPLSE
      WD(2)=PNVARY*RNDM(2)*TO
      WD(3)=PNVARZ*RNDM(3)*TO
C
```

```
C
C     *** COMPUTE BNWD°200 =BN°200 X 3  * WD°3
C
      DO 5035 I=NS,NF
         SUM=0.0
             DO 5030 K=1,3
             SUM=SUM+BN(I,K)*WD(K)
5030         CONTINUE
         BNWD(I)=SUM
5035  CONTINUE
C
C     *** COMPUTE X°200 =AX°200  + BNWD°200
C
      DO 5040 I=NS,NF
         X(I)= AX(I) + BNWD(I)
         IF (DABS(X(I)).LT. 1.0D-60) THEN
             X(I)=1.0D-60
         END IF
C
C
C
5040  CONTINUE
C
C     *** COMPUTE V°3
C
      V(1)=MNVARX*RNDM(4)
      V(2)=MNVARY*RNDM(5)
      V(3)=MNVARZ*RNDM(6)
C
C     *** COMPUTE Y°3  = C°3 X 200  * X°200  + V°3
C
      DO 5050 I=1,3
         SUM=0.0
             DO 5045 K=NS,NF
             SUM=SUM+C(I,K)*X(K)
5045         CONTINUE
         Y(I)=SUM+V(I)
5050  CONTINUE
C
C     ***********************
C
C     *** START OF KALMAN FILTER ***
C
      M2=2*MODE
C
C
C     *** COMPUTE XH1 = A*XH
C
      DO 300 I=JP,JQ
      SUM=0.0
         DO 295 K=JP,JQ
         SUM=SUM+A(I,K) * XH(K)
295      CONTINUE
      XH1(I)=SUM
300   CONTINUE
C     ****************
```

```
C
C     *** COMPUTE DY = Y - C*XH1
C
      DO 315 I=1,3
      SUM=0.0
         DO 310 K=JP,JQ
         SUM=SUM+C(I,K)*XH1(K)
310      CONTINUE
      DY(I)=Y(I)-SUM
315   CONTINUE
C
C     ******************
C
C     *** COMPUTE XH = XH1 + G*DY
C
      DO 325 I=1,M2
         J1=JP-1+I
         SUM=0.0
             DO 320 K=1,3
             SUM=SUM+G(I,K)*DY(K)
320          CONTINUE
         XH(J1)=XH1(J1)+SUM
         IF (DABS(XH(J1)).LT.1.0D-60) THEN
             XH(J1)=1.0*D-60
         END IF
C
325   CONTINUE
C
C
C
C     ***** END OF KALMAN ROUTINES ******
C
C     *** COMPUTATION OF ESUM ***
      DO 340 I=JP,JQ
      XDEL= X(I)-XH(I)
      XDEL1=XDEL*XDEL*SAMPT
      E2(I)=E2(I)+XDEL1
      XS(I)=XS(I)+X(I)*X(I)*SAMPT
      E3(I)=E2(I)/XS(I)
340   CONTINUE
C
      CGN=CGN+1.0
      IF (CTT.EQ.1.0.OR.CGN.GT.PRT) THEN
C
      WRITE (6,*) 'TIME= ', TIME, ' SEC.'
C
      WRITE (17,1008)
      WRITE (16,1008)
      WRITE (16,2100) TIME
C
      WRITE (17,2100) TIME
2100  FORMAT(' ','TIME=  ',F9.3)
      DO 380 I=JP , JQ
      WRITE (16,4500) I,X(I) ,I ,XH(I)
C
```

```
      WRITE (17,4530) I,E2(I)  ,E3(I)   , XS(I)
380   CONTINUE
C
C
C
C
      CGN=0.0
      END IF
4500  FORMAT (' ',' X(',I3,')= ',D15.8,2X,'XH(',I3,')= ',D15.8)
4530  FORMAT (' ',5X,I5,5X,3 D15.8)
C
400   CONTINUE
C
C
      DO 401 I=JP,JQ
         WRITE (19,4530) I, E2(I) ,E3(I), XS(I)
401   CONTINUE
C
C
C
C
599   STOP
      END
C
C
C     ******************************************************************
C     THIS SUBROUTINE COMPUTES THE STATE TRANSITION MATRIX FOR EACH
C     OF THE 100 MODES
C     ******************************************************************
C
      SUBROUTINE STMTRX(EMODE,SMODE,T,D,PHI,GAMMA,A,B,LAMA,UGVEX,C,
     +         ROWN1,ROWN2,ROWN3,BN)
C
      REAL*8 WN,GMA,PHI(2,2,100),GAMMA(2,100),EGT,T,COSW1T,SINW1T
      REAL*8 W1,D,A(200,200),B(200,3),C(3,200),BN(200,3)
      REAL LAMA(100),UGVEX(684,100)
      INTEGER SMODE,R,EMODE,JJ,KK,ROWN1,ROWN2,ROWN3
C
C
      DO 600 I = 1    ,100
         WN = DBLE(SQRT(LAMA(I)))
         GMA = D*WN/2.0
         EGT = DEXP(-GMA*T)
         W1 = DSQRT((WN**2)-(GMA**2))
         COSW1T = DCOS(W1*T)
         SINW1T = DSIN(W1*T)
C
C
C
C
         IF(WN.EQ.0)THEN
              PHI(1,1,I) = EGT*COSW1T
              PHI(1,2,I) = T
              PHI(2,1,I) = 0
              PHI(2,2,I) = EGT*COSW1T
C
```

```
C
C
C
C
C
                GAMMA(1,I) = 0
                GAMMA(2,I) = 0
           ELSE
C
C
C
C
           PHI(1,1,I) = EGT*(COSW1T + (GMA*(W1**(-1)))*SINW1T)
           PHI(1,2,I) = (W1**(-1))*EGT*SINW1T
           PHI(2,1,I) = -(WN**2)*(W1**(-1))*EGT*SINW1T
           PHI(2,2,I) = EGT*(COSW1T - (GMA*(W1**(-1)))*SINW1T)
C
C
C
C
        GAMMA(1,I)=(WN**(-2))*(1.0D0-EGT*COSW1T -EGT*(GMA/W1)*SINW1T)
           GAMMA(2,I) = (W1**(-1))*EGT*SINW1T
C
C
C
C
           ENDIF
C
C
C
600     CONTINUE
C
C
        R = 1
C
        DO 610 K = 1    ,100
C
C
C
C
C
           A(R,R) = PHI(1,1,K)
           A(R,R+1) = PHI(1,2,K)
           A(R+1,R) = PHI(2,1,K)
           A(R+1,R+1) = PHI(2,2,K)
C
C
C
C
C
C       *** B MATRIX FOR MULTIPLYING CONTROL TORQUES
C
           B(R,1) = GAMMA(1,K)*DBLE(UGVEX(412,K))
           B(R,2) = GAMMA(1,K)*DBLE(UGVEX(413,K))
           B(R,3) = GAMMA(1,K)*DBLE(UGVEX(414,K))
           B(R+1,1) = GAMMA(2,K)*DBLE(UGVEX(412,K))
```

```
         B(R+1,2) = GAMMA(2,K)*DBLE(UGVEX(413,K))
         B(R+1,3) = GAMMA(2,K)*DBLE(UGVEX(414,K))
C
C
C
C
C
C
C
C      *** BN MATRIX FOR MULTIPLYING THE NOISE DISTURBANCES
C
C
C
C
      BN(R,1)=GAMMA(1,K)*DBLE(UGVEX(ROWN1,K))
      BN(R,2)=GAMMA(1,K)*DBLE(UGVEX(ROWN2,K))
      BN(R,3)=GAMMA(1,K)*DBLE(UGVEX(ROWN3,K))
      BN(R+1,1)=GAMMA(2,K)*DBLE(UGVEX(ROWN1,K))
      BN(R+1,2)=GAMMA(2,K)*DBLE(UGVEX(ROWN2,K))
      BN(R+1,3)=GAMMA(2,K)*DBLE(UGVEX(ROWN3,K))
C
C
C
C
C
C
         R = R+2
610   CONTINUE
C
C      *** C MATRIX PRODUCTION ***
C
C
C
C
      JJ=-1
      DO 640 I=1,100
      JJ=JJ+1
      KK=I        +JJ
C
C
C
      C(1,KK) = DBLE(UGVEX(418,I))
      C(2,KK) = DBLE(UGVEX(419,I))
      C(3,KK) = DBLE(UGVEX(420,I))
C
C
C
      KK=KK+1
C
      C(1,KK)=0.0
      C(2,KK)=0.0
      C(3,KK)=0.0
C
640   CONTINUE
C
C
```

```
C                                                                       SIM06610
C                                                                       SIM06620
      RETURN                                                            SIM06630
      END                                                               SIM06640
```

APPENDIX C. PROGRAM TO ESTIMATE NOISE IN KALMAN FILTER FROM UNOBSERVED MODES

```
C
C     ***********************************************************
C     *****                SPAC 24                          *****
C     *****  ADAPTED TO RUN N MODES OF THE PLANT AND        *****
C     *****  COMPUTE THE NOISE IN THE KALMAN FILTER         *****
C     *****  FROM THE UNOBSERVED MODES                      *****
C     ***********************************************************
C
C
C     ***********************************************************
C     *****           VARIABLE DECLARATIONS                 *****
C     ***********************************************************
C
      EXTERNAL STMTRX,EXCMS
      CHARACTER*6 NAM
      CHARACTER*1 AGAIN,CORECT,RAGAIN
      INTEGER ROWN1,ROWN2,ROWN3,COUNT,NODE,MODE,KQ,EMODE,SMODE,R2M,C2M
      INTEGER CT,CF,KADJ,CFADJ,LOOP,PRNT,JJ,JK,N1,JR,KR,MR,ISEED,M2
      INTEGER NO,                    NS, NF, SN, FN
      INTEGER JL,J1,JM , JP, JQ, KA, KB, KC, KD, KE, KF, KG
C
C
C
      REAL LAMA(100), UGVEX(684,100),RNODE,RMODE,MIN
      REAL*8 PHI(2,2,100),GAMMA(2,100),EGT,GMA,WN,W1,X1T,X2T,TIME
      REAL*8 A(200,200),B(200,3),F(3, 50),IMPLSE,ENERGY
      REAL*8 COSW1T,SINW1T,X(200)
      REAL*8 DAMP,SAMPT,PI,SAMPTM,SUM1,SUM2,SUM3,SUMC
      REAL*8 C(9,200), RMN(3,3), QPN(3,3)
      REAL*8 BN(200,3)
      REAL*8 PNVARX, PNVARY, PNVARZ
      REAL*8 MNVARX, MNVARY, MNVARZ
      REAL*8 SUM, RNDM(6), RND1, RND2
      REAL*8 ES,ED,ESUM,CGN,PRT
      REAL*8 WT , WD(3),  BNWD(200), EX1(9)
      REAL*8 EX(9),AX(200) , SF , TO, CTT, ESS
      REAL*8 CTG, XDEL,  XDEL1, ERS, PRT1
      REAL*8 SF1
C
C     ***********************************************************
C     *****           VARIABLE DEFINITIONS                  *****
C     ***********************************************************
C
C     STMTRX = SUBROUTINE EXTABLISHES STATE TRANSITION MATRICIES
C     LAMA = VECTOR OF THE SQUARE OF THE NATURAL FREQUENCIES
C     UGVEX = MODE POSITONS AND SLOPES OF THE NODAL POINTS
C     PHI = STATE TRANSITION MATRICIES FOR EACH MODE
C     GAMMA = INPUT TRANSITION MATRIX
C     A = DIAGONAL MATRIX CONSISTING OF PHI
```

```
C      B = INPUT MATRIX OF GAMMA AND CONTROL SLOPES
C      DAMP = DAMPING FACTOR
C      SAMPT = SAMPLING TIME
C      IMPLSE = IMPULSE INPUT FUNCTION
C      MIN = NUMBER OF MINUTES SYSTEM WILL BE OBSERVED
C      SMODE = NUMBER OF STARTING MODE (INT)
C      MODE = NUMBER OF MODES (INT)
C      EMODE = NUMBER OF THE LAST MODE (INT)
C      NODE = NUMBER OF THE NOISE INPUT MODE (INT)
C       *** NOISE SLOPE LOCATIONS IN DATA MATRIX ***
C      ROWN1 = X-SLOPE LOCATION
C      ROWN2 = Y-SLOPE LOCATION
C      ROWN3 = Z-SLOPE LOCATION
C      C = OUTPUT MATRIX FOR Y
C      IDENT = IDENTITY MATRIX
C      RMN = MEASUREMENT NOISE COVARIANCE MATRIX
C      QPN = PLANT NOISE COVARIANCE MATRIX
C      PNVARX = PLANT NOISE X-SLOPE VARIANCE
C      PNVARY = PLANT NOISE Y-SLOPE VARIANCE
C      PNVARZ = PLANT NOISE Z-SLOPE VARIANCE
C      MNVARX = MEASUREMENT NOISE X-SLOPE VARIANCE
C      MNVARY = MEASUREMENT NOISE Y-SLOPE VARIANCE
C      MNVARZ = MEASUREMENT NOISE Z-SLOPE VARIANCE
C      ISEED = INITIALIZATION FOR RANDOM NUMBER GENERATOR
C      RNDM = RANDOM NUMBERS USED FOR WHITE NOISE IN MEASUREMENTS AND
C              IN PLANT FORCES
C      BN = B MATRIX TO MULTIPLY NOISE DISTURBANCES
C
C
C
C      **********      SAMPLE OF SPACE EXEC FILE     *******************
C
C      THIS FILE MUST BEGIN IN COLUMN 1 AND RUN WITH THE FOLLOWING
C      SEQUENCE FOR THE INITIAL RUN OF THE PROGRAM:
C
C              FORTVS SPACE          (COMPILES PROGRAM)
C              SPACE                 (EXECUTES EXEC FILE)
C              LOAD SPACE (START     (LOADS AND EXECUTES PROGRAM)
C
C      SUBSEQUENT PROGRAM RUNS CAN ELIMINATE "FORTVS SPACE" IF NO
C      CHANGES HAVE BEEN MADE TO THE PROGRAM, AND CAN ELIMINATE
C      RUNNING THE EXEC FILE.
C
C      FI 4 DISK THESIS INPUT (PERM
C      FI 8 DISK UTILITY DATA  (RECFM VS BLOCK 133 PERM
C      FI 11 DISK CNTRL OUTPUT  (RECFM F BLOCK 80 LRECL 80 PERM
C      FI 13 DISK GAMMA OUTPUT  (RECFM VS BLOCK 133 PERM
C      FI 14 DISK MODE OUTPUT  (RECFM F BLOCK 80 LRECL 80 PERM
C      FI 16 DISK COST OUTPUT  (RECFM F BLOCK 80 LRECL 80 PERM
C      FI 17 DISK PRT  OUTPUT  (RECFM F BLOCK 80 LRECL 80 PERM
C      FI 18 DISK ERROR DATA   (RECFM F BLOCK 80 LRECL 80 PERM
C      FI 19 DISK END FILE   (RECFM F BLOCK 80 LRECL 80 PERM
C      FI 20 DISK GMAT FILE (RECFM F BLOCK 80 LRECL 80 PERM
C
C      *****************************************************************
C
```

```
      PARAMETER (JR=5243, KR=5397, MR=262139)
C
C
      MIN = 15.0
C
      WT=1.0D00
      PI = 4.0D0 * ATAN(1.0D0)
C
C     ******************************************************************
C     *****           READ LAMA AND UGVEX MATRICIES               *****
C     ******************************************************************
C
      CALL EXCMS ('CLRSCRN')
      WRITE(6,1008)
      WRITE(6,*) '                       READING LAMA AND UGVEX MATRICIES'
      WRITE(6,*) ' '
C     THIS SECTION READS THE LAMA VECTOR AND THE UGVEX
C     MATRIX AND STORES THEM IN MEMORY FOR FURTHER RECALL OF
C     DESIRED LOCATION DATA.
C
      READ(4,1001) NAM
      READ(4,1002)(LAMA(I),I=1,100)
      READ(4,1001) NAM
      DO 5 J = 1,100
         READ(4,1002)(UGVEX(I,J),I=1,684)
5     CONTINUE
C
1001  FORMAT(1X,A6)
1002  FORMAT(1X,8E15.8)
1008  FORMAT(1X,////)
C
500   CALL EXCMS ('CLRSCRN')
C
C     ************        STARTING MODE NUMBER        **************
C     ** SMODE 7 TO 100 (INTEGER) ****
      SMODE= 17
C
      WRITE (16,700) SMODE
700   FORMAT (' ','STARTING MODE NUMBER:  ',I2)
C
C     ***************     NUMBER OF MODES TO SCAN     **************
C     ** MODE 1 TO 93 (INTEGER)
C
      MODE=3
C
      EMODE = SMODE + MODE - 1
C
      WRITE (16,701) MODE
701   FORMAT (' ','NUMBER OF MODES SCANNED:  ',I2)
C
C     ************     NOISE INPUT POSITION      *****************
C     ** NODE 1 TO 114 (INTEGER) (IF 0 THEN NO NOISE INPUT)
      NODE= 8
C
      WRITE (16,702) NODE
702   FORMAT (' ',' NOISE NODE LOCATION:   ',I5)
```

```
C
C     ************ START AND STOP FOR PLANT
      SN=17
      FN=4
      NS=SN*2-1
      NF=SN*2+2*FN-2
      WRITE (16,899) SN,FN
899   FORMAT (' ','PLANT -- SN= ',I5,'  FN= ',I5)
C     **************          SAMPLING TIME          **************
C     ** SAMPT MUST BE LESS THAN OR EQUAL TO SAMPTM **
      SAMPT = 0.05
         SAMPTM = ((2.0D0*PI)/SQRT(LAMA(EMODE)))/1.0D01
         IF (SAMPT.GE.SAMPTM) THEN
             SAMPT=SAMPTM
         ENDIF
C
      WRITE (16,900) MIN
900   FORMAT (' ',2X,'MIN:  ',F8.3)
C
      WRITE (16,703) SAMPT, SAMPTM
703   FORMAT (' ','SAMPLING TIME:  ',D12.4,2X,'SAMPTM= ',D15.8)
C
C     ****************          DAMPING FACTOR          **************
C     ** DAMP 0.0 TO 1.0 (REAL*8)
      DAMP=.01
C
      WRITE (16,704) DAMP
704   FORMAT (' ','DAMPING FACTOR:  ',D12.4)
C
      NO=3
C     *** PLANT NOISE VARIANCE ***
C     ** PNVARX, PNVARY, PNVARZ GT 0.0
C
      SF=1.0D0
      SF1=2.5D06
      PNVARX=1.0D00*SF1
      PNVARY=1.0D00*SF1
      PNVARZ=1.0D00*SF1
C
C
C
C     *** MEASUREMENT NOISE VARIANCE ***
C     ** MNVARX, MNVARY, MNVARZ GT 0.0
      MNVARX=1.0D-03 *SF
      MNVARY=1.0D-03 *SF
      MNVARZ=1.0D-03 *SF
C
C
      WRITE (16,711)
711   FORMAT(' ','PLANT NOISE VARIANCE:  ')
      WRITE (16,712)
712   FORMAT(' ',6X,'PNVARX',13X,'PNVARY',13X,'PNVARZ')
      WRITE (16,713) PNVARX, PNVARY, PNVARZ
713   FORMAT(' ',2X,E15.8,2X,E15.8,2X,E15.8)
      WRITE(16,714)
714   FORMAT(' ','MEASUREMENT NOISE:')
```

```
      WRITE(16,715)
715   FORMAT(' ',6X,'MNVARX',13X,'MNVARY',13X,'MNVARZ')
      WRITE(16,713) MNVARX,MNVARY,MNVARZ
C
510   CALL EXCMS ('CLRSCRN')
      WRITE (6,1008)
      WRITE (6,*) '                              PROGRAM RUNNING'
C
C     ***********       NOISE INPUT LOCATION        ************
C
      ROWN3 = NODE*6
      ROWN2 = (NODE*6) - 1
      ROWN1 = (NODE*6) - 2
      COUNT = 0
C
C     ***********      INITIALIZE MATRICIES       ****************
C
      DO 54 K = 1, 200
          X(K) = 0.0
54    CONTINUE
C
      DO 60 I=1,3
         DO 58 J=1,3
         RMN(I,J)=0.0
         QPN(I,J)=0.0
58       CONTINUE
60    CONTINUE
C
      RMN(1,1)=MNVARX**2.0
      RMN(2,2)=MNVARY**2.0
      RMN(3,3)=MNVARZ**2.0
      QPN(1,1)=PNVARX**2.0
      QPN(2,2)=PNVARY**2.0
      QPN(3,3)=PNVARZ**2.0
C
C     ***********************************************************
C     *****             BEGIN MAIN PROGRAM                  *****
C     ***********************************************************
C
      CALL STMTRX(EMODE,SMODE,SAMPT,DAMP,PHI,GAMMA,A,B,LAMA,UGVEX,C,
     +   ROWN1,ROWN2,ROWN3,BN)
C
C
      WRITE (6,1008)
      WRITE(6,*) ' EXIT STMTRX - - - PRE-LOOP KALMAN'
C
C
      WRITE (6,*) 'COMPUTING C TIMES SF FOR NEW C'
C
      WRITE (16,1008)
      DO 11000 I=1,200
         DO 10900 J=1,NO
         C(J,I)= C( J,I)*SF
10900    CONTINUE
11000 CONTINUE
C
```

```
C     *** PRE-LOOP PORTION OF KALMAN FILTER
C
C
      JK=SMODE*2-2
      M2=2*MODE
C
C
C     ******************************
C
      M2=2*MODE
      JP=2*SMODE-1
      JQ=2*EMODE
C
C
      DO 8813 I=1,3
      EX(I)=0.0
8813  CONTINUE
C
C
C
C
C     ***********************************************************
C     *****    THIS SECTION COMPUTES THE STATE UPDATE      *****
C     ***********************************************************
      ESS =0.0
      COUNT = 0
      ENERGY = 0.0D0
      TIME = 0.0
      CGN=0.0
      CTG=0.0
C     *****  SETS LOOP FOR THE ITERATIONS NECESSARY TO OBSERVE  *****
C     *****  THE SYSTEM FOR THE NUMBER OF MINUTES SPECIFIED     *****
      WRITE (6,1008)
      WRITE (6,*) '            START STATE UPDATE    '
      LOOP = INT((MIN*60.0)/SAMPT)
      PRT= (DBLE(LOOP))/30.0
      PRT1=(DBLE(LOOP))/50.00
      CTT=0.0
C
      DO 400 L  = 0, LOOP
         TIME = DBLE(L)*SAMPT
C
         IF(L.EQ.0)THEN
            IMPLSE =(1.0D06*SF1)/(DSQRT(SAMPT))
         ELSE
            IMPLSE = 0.0D0
         ENDIF
C
      TO=0.0
C     ***** RANDOM NUMBER GENERATOR *****
C
      DO 101 I=1,6
      ISEED=MOD(ISEED*JR+KR,MR)
      RND1=(DBLE(ISEED)+0.5D00)/DBLE(MR)
      ISEED=MOD(ISEED*JR+KR,MR)
      RND2=(DBLE(ISEED)+0.5D00)/DBLE(MR)
```

```
      RNDM(I)=DSQRT(-2.0*DLOG(RND1))*DCOS(6.2831853D00*RND2)
101   CONTINUE
C     *******************
      CTT=CTT+1.0
C     **** START OF STATE UPDATE ***
C
C     *** COMPUTE AX°200  = A°200 X 200  * X°200
C
C
C     *** COMPUTE AXH = A*XH
C
      JK=SMODE*2-2
      JP=JK+1
      JQ=2*EMODE
C
C
      DO 5015 I=NS,NF
         SUM=0.0
              DO 5010 K=NS,NF
              SUM=SUM+A(I,K)*X(K)
5010          CONTINUE
         AX(I)=SUM
5015  CONTINUE
C
C
C     *** COMPUTE WD°3
C
      WD(1)=PNVARX*RNDM(1)*TO+IMPLSE
      WD(2)=PNVARY*RNDM(2)*TO
      WD(3)=PNVARZ*RNDM(3)*TO
C
C     *** COMPUTE BNWD°200 =BN°200 X 3  * WD°3
C
      DO 5035 I=NS,NF
         SUM=0.0
              DO 5030 K=1,3
              SUM=SUM+BN(I,K)*WD(K)
5030          CONTINUE
         BNWD(I)=SUM
5035  CONTINUE
C
C     *** COMPUTE X°200 =AX°200  + BNWD°200
C
      DO 5040 I=NS,NF
         X(I)= AX(I) + BNWD(I)
         IF (DABS(X(I)).LT. 1.0D-60) THEN
              X(I)=1.0D-60
         END IF
C
5040  CONTINUE
C
C
C     *******************
C
C     *** START OF KALMAN FILTER ***
C
```

```
      JK=SMODE*2-2
      JP=JK+1
      JQ=2*EMODE
      M2=2*MODE
C
      JL=JQ+1
      DO 8888 I=1,NO
      SUM=0.0
         DO 8887 K=JL,NF
         SUM=SUM+C(I,K)*X(K)
8887     CONTINUE
      EX(I)=SUM*SUM*SAMPT+EX(I)
8888  CONTINUE
C
C
      CGN=CGN+1.0
      IF (CTT.EQ.1.0.OR.CGN.GT.PRT) THEN
C
      WRITE (16,1008)
      WRITE (16,*)   'TIME = ', TIME
C
      DO 380 I=JP , JQ
      WRITE (16,4500) I,X(I)
380   CONTINUE
4500  FORMAT (' ',2X,'X(',I4,')= ',D15.8)
C
      CGN=0.0
      END IF
C
C
400   CONTINUE
      WRITE (11,*) 'SMODE = ', SMODE
      WRITE (11,*) 'EMODE = ', EMODE
      WRITE (11,*) 'SN = ',SN
      WRITE (11,*) 'FN = ',FN

C
      JL=JQ+1
      DO 9499 I=1,NO
      WRITE (11,*) 'EX  ',I ,'      ', EX(I)
9499  CONTINUE
C
C
C
C
C
      CALL EXCMS ('CLRSCRN')
      WRITE (6,1008)
C
599   STOP
      END
C
C
C
C
C
```

```
C     **********************************************************************
C     THIS SUBROUTINE COMPUTES THE STATE TRANSITION MATRIX FOR EACH
C     OF THE 100 MODES
C     **********************************************************************
C
      SUBROUTINE STMTRX(EMODE,SMODE,T,D,PHI,GAMMA,A,B,LAMA,UGVEX,C,
     +         ROWN1,ROWN2,ROWN3,BN)
C
      REAL*8 WN,GMA,PHI(2,2,100),GAMMA(2,100),EGT,T,COSW1T,SINW1T
      REAL*8 W1,D,A(200,200),B(200,3),C(9,200),BN(200,3)
      REAL LAMA(100),UGVEX(684,100)
      INTEGER SMODE,R,EMODE,JJ,KK,ROWN1,ROWN2,ROWN3, NN(9), N9, NO
C
C
      WRITE (6,*) 'INSIDE STMTRX - - COMPUTE WN, GMA, EFT, W1'
C
C
      DO 600 I = 1    ,100
         WN = DBLE(SQRT(LAMA(I)))
         GMA = D*WN/2.0
         EGT = DEXP(-GMA*T)
         W1 = DSQRT((WN**2)-(GMA**2))
         COSW1T = DCOS(W1*T)
         SINW1T = DSIN(W1*T)
C
C
C
C
         IF(WN.EQ.0)THEN
              PHI(1,1,I) = EGT*COSW1T
              PHI(1,2,I) = T
              PHI(2,1,I) = 0
              PHI(2,2,I) = EGT*COSW1T
C
C
C
C
C
C
              GAMMA(1,I) = 0
              GAMMA(2,I) = 0
         ELSE
C
C
C
C
         PHI(1,1,I) = EGT*(COSW1T + (GMA*(W1**(-1)))*SINW1T)
         PHI(1,2,I) = (W1**(-1))*EGT*SINW1T
         PHI(2,1,I) = -(WN**2)*(W1**(-1))*EGT*SINW1T
         PHI(2,2,I) = EGT*(COSW1T - (GMA*(W1**(-1)))*SINW1T)
C
C
C
C
      GAMMA(1,I)=(WN**(-2))*(1.0D0-EGT*COSW1T -EGT*(GMA/W1)*SINW1T)
```

```
         GAMMA(2,I) = (W1**(-1))*EGT*SINW1T
C
C
C
C
         ENDIF
C
C
C
600   CONTINUE
C
      WRITE (6,*) 'PHI AND GAMMA COMPUTED'
      WRITE (6,*) ' COMPUTING A, B, BN'
C
      R = 1
C
      DO 610 K = 1    ,100
C
C
C
C
C
         A(R,R) = PHI(1,1,K)
         A(R,R+1) = PHI(1,2,K)
         A(R+1,R) = PHI(2,1,K)
         A(R+1,R+1) = PHI(2,2,K)
C
C
C
C
C
C     *** B MATRIX FOR MULTIPLYING CONTROL TORQUES
C
         B(R,1) = GAMMA(1,K)*DBLE(UGVEX(412,K))
         B(R,2) = GAMMA(1,K)*DBLE(UGVEX(413,K))
         B(R,3) = GAMMA(1,K)*DBLE(UGVEX(414,K))
         B(R+1,1) = GAMMA(2,K)*DBLE(UGVEX(412,K))
         B(R+1,2) = GAMMA(2,K)*DBLE(UGVEX(413,K))
         B(R+1,3) = GAMMA(2,K)*DBLE(UGVEX(414,K))
C
C
C
C
C
C
C
C     *** BN MATRIX FOR MULTIPLYING THE NOISE DISTURBANCES
C
C
C
C
      BN(R,1)=GAMMA(1,K)*DBLE(UGVEX(ROWN1,K))
      BN(R,2)=GAMMA(1,K)*DBLE(UGVEX(ROWN2,K))
      BN(R,3)=GAMMA(1,K)*DBLE(UGVEX(ROWN3,K))
      BN(R+1,1)=GAMMA(2,K)*DBLE(UGVEX(ROWN1,K))
      BN(R+1,2)=GAMMA(2,K)*DBLE(UGVEX(ROWN2,K))
```

```
      BN(R+1,3)=GAMMA(2,K)*DBLE(UGVEX(ROWN3,K))
C
C
C
C
C
C
        R = R+2
610   CONTINUE
C
      WRITE (6,*) 'A, B, BN COMPUTED'
C
      WRITE (6,*) 'COMPUTING C'
C
C
C
C     *** C MATRIX PRODUCTION ***
      NO=3
      NN(1)=418
      NN(2)=419
      NN(3)=420
C
C
C
C
C
C
      JJ=-1
      DO 640 I=1,100
      JJ=JJ+1
C
      DO 9127 K=1,NO
C
      KK=I+JJ
C
      N9=NN(K)
C
      C(K,KK) = DBLE(UGVEX(N9,I))
C
      KK=KK+1
C
      C(K,KK)=0.0
9127  CONTINUE
640   CONTINUE
C
C
C
      RETURN
      END
```

LIST OF REFERENCES

1. Rodriguez, G., *Control Technology Development*, Large Space Systems Technology - 1980, NASA Conference Publication 2168, Vol. I, pp. 49-63, November 18-20, 1980.

2. Thomson, William T., *Theory of Vibration with Applications*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1988.

3. Preston, William J. , *Effects of Reduced Order Modeling on the Control of a Large Space Structure*, Masters Thesis, Naval Postgraduate School, CA, September 1988.

4. Selby, Samuel M., *Standard Mathematical Tables*, The Chemical Rubber Co., Cleveland, OH, 1972.

5. Mendel, Jerry M., *Discrete Techniques of Parameter Estimation*, Marcel Dekker, Inc., NY, 1973.

6. Nishimura, T., "On the a priori Information in Sequential Estimation Problems," *Kalman Filtering: Theory and Aplication*, p. 138-145, Edited by: Sorenson, H. W., IEEE Press, Inc., New York, 1985.

7. Franklin, Gene F. and Powell, David J., *Digital Control of Dynamic Systems*, Addison-Wesley Company, Reading, MA, 1980.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, CA 93943-5002	2
3.	Director Naval Research Laboratory Washington, D.C. 20375	1
4.	Commander Naval Space Command Attn: Code N3 Dahlgren, VA 22448-5170	1
5.	Chairman, Code 62 Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943	1
6.	Prof. Jeff B. Burl, Code 62BL Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943	1
7.	Prof. H. A. Titus, Code 62TS Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943	1
8.	Major W. J. Preston 7 Cary Place Fredricksburg, VA 22405	1
9.	Lcdr. Bruce M. Jackson c/o Col. Jose A. Aybar (USA-Ret) 22577 Veronica Drive Salinas, CA 93943	2

Thesis
J2125 Jackson
c.1 Utilization of a Kal-
man observer with large
space structures.